Marzo 1983 - Anno III nº 2 - Sped. Abb. Post. Gr. III/70 - Distr. MePe

Vinci $ 170.000 con Commodore Computer Club

Esclusivo: Le tecniche di overlay

Memorie di massa e stampante

Programmi e giochi per Vic 20 e Commodore 64

FINALMENTE.
LA TAVOLETTA GRAFICA A PIENE PRESTAZIONI AD UN PREZZO ACCESSIBILE A TUTTI

Koala
Disponibile per Apple][+ e //e
Atari 400 e 800, Commodore
ed IBM P.C.

La tavoletta grafica KOALA è la più simpatica innovazione nel campo dei personal computers. Con KOALA, controllate il vostro computer con un dito. Più veloce di un paddle, più versatile di un joystick e più semplice di una tastiera.

La tavoletta grafica KOALA è compatibile con la maggior parte di software esistente e viene fornita completa del suo programma grafico "Micro Illustrator". KOALA-PAD è il miglior modo per creare immagini ad alta risoluzione con il vostro computer.

COMPUTER GRAPHICS DIVISION

MILANO: Via L. da Vinci, 43 - 20090 Trezzano S/N
Tel. 02/4455741/2/3/4/5 - Tlx: TELINT I 312827

ROMA: Via Salaria, 1319 - 00138 Roma
Tel. 06/6917058-6919312 - Tlx: TINTRO I 614381

SOMMARIO

Direttore responsabile: Michele di Pisa
Redattore capo: Alessandro De Simone
Redazione/collaboratori: Giovanni Bellù, Maurizio Di Vizio
Direzione, redazione: V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
Pubblicità: Mirco Croce (coordinatore), Paola Bevilacqua, Michela Prandini, Giorgio Ruffoni, Marco Ravagli, Roberto Sghirinzetti, Claudio Tidone, Villa Claudio - V.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348/9/40
Prezzo e abbonamenti: Prezzo per una copia Lire 2.500
Arretrati il doppio. Abbonamento per dieci fascicoli lire 18.000
Abbonamento annuo cumulativo alle riviste Computer e Commodore Computer Club (tariffa riservata agli studenti) L. 34.000. I versamenti vanno indirizzati a: Commodore Computer Club, mediante assegno bancario, vaglia o utilizzando il c/c postale n. 31532203
Composizioni: Minisystems Italia
Selezioni: Org. A.G.
Stampa: La Litografica s.r.l. - Busto A.
Registrazione: Tribunale di Milano n. 2/10/1982 N.ro 370. Sped. in abb. post. gr. III. Pubbl. inferiore al 70%

SERPENTE

Un simpatico serpente appena nato, lungo appena alcuni quadretti video, comandato dal giocatore, si aggira per un labirinto in cerca di cibo, delle piccole bisce, e, ogni volta che ne mangia una, cresce e diventa sempre più lungo. Scopo del gioco è di mangiare tutte le bisce; ma attenzione: il serpente è velenoso, e se morsica sè stesso, cioè si intrappola con la stessa coda, muore ed il gioco finisce.

Chi riesce a mangiare tutte le bisce, passa ad un altro labirinto, una specie di reticolo, col quale aumenta il livello di difficoltà. Se, cioè, prima d'ogni biscia mangiata il serpente si allungava di due, adesso si allunga di tre, mentre il numero delle bisce cresce. Se anche in questo labirinto riuscite a mangiare tutte le bisce, si ritorna al primo, ma con un numero di bisce ancora maggiore, mentre il serpente questa volta si allunga di quattro per ogni biscia mangiata.

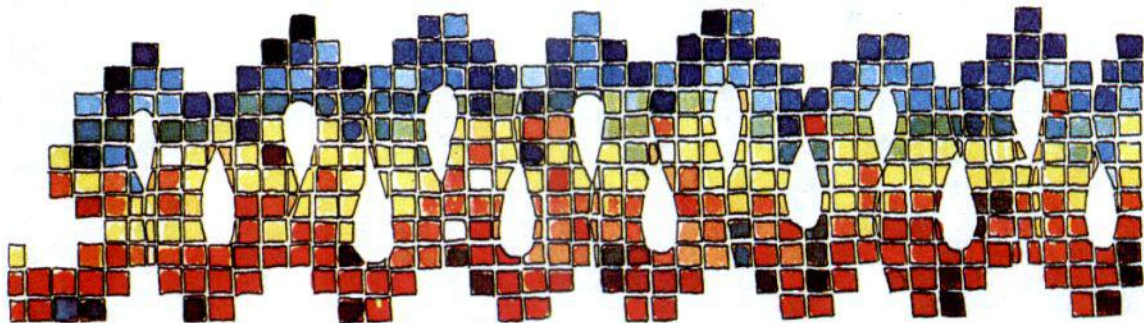
Il gioco prosegue alternando il primo ed il secondo labirinto, ed aumentando sempre più il livello di difficoltà, fino a quando il serpente diventerebbe troppo lungo da non poter più stare nel labirinto. A questo punto si ritorna a livello zero.

Vi assicuro però che questo non è facile, anzi... fra parentesi dico che io non ci sono mai riuscito, ma non è mai detta l'ultima parola.

Come si usa

All'inizio bisogna aspettare circa un minuto per permettere al computer di rilocare la mappa caratteri. Attenzione: durante questa fase è disabilitata la tastiera e nemmeno usando Run Stop+Restore, si riesce a bloccare il programma. Perciò registrate il programma prima di farlo girare, perchè un eventuale errore di copiatura potrebbe bloccare il sistema, con la perdita del programma appena copiato.

In seguito bisogna dare il pro-

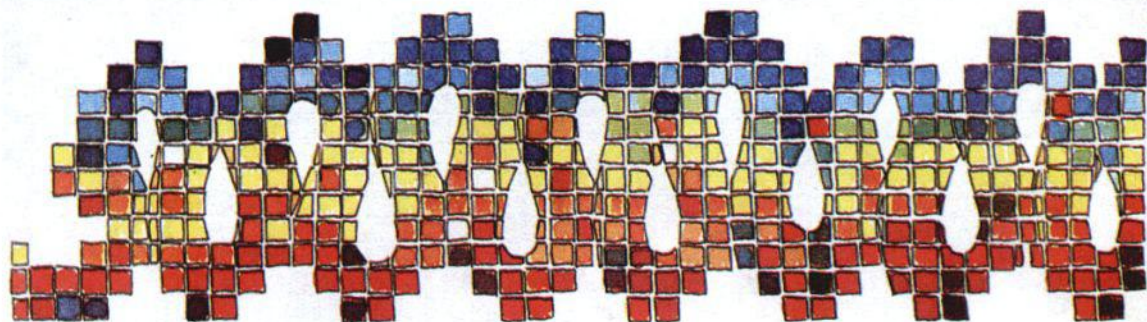
prio nome, che verrà stampato in alto a destra. Altre informazioni che vengono visualizzate sulla parte destra del video sono il tempo, il numero delle bisce mangiate; il livello di difficoltà e i relativi parametri di numero bisce e di aumento per biscia.

All'inizio sembra che il serpente vada dove vuole: non spaventatevi perchè quando c'è da fare una curva obbligata, il computer non aspetta che voi cambiate direzione, ma lo fa lui automaticamente, per rendere il gioco più difficile. Si può incontrare però anche un bivio, cioè un incrocio con due possibilità di direzione: in questo caso si ferma ed attende che voi indichiate dove volete andare. Chi volesse rendere il gioco ancora più difficile, potrebbe modificare il programma in modo da far decidere al computer la direzione anche a un bivio. Altre sofisticazioni del programma sono possibili: per esempio, si possono mettere più labirinti, ognuno diverso dall'altro ed in ordine crescente di difficoltà...

Note

Il programma così come è pubblicato, funziona solo mettendo un joystick nella porta uno. Le direzioni consentite sono: alto, basso, destra e sinistra. Se non si ha a disposizione un joystick, bisogna apportare al programma alcune modifiche. ▪

```
100 POKE53280,0:POKE53281,2:PRINT"■":GOSUB1580
110 PRINT"■":CLR:Q=54273:POKE54296,15:POKE54277,190:DIMA(350),A$(5,1)
:Z=2:NR=15
120 E$="■":FORK=1TO17:E$=E$+"■":NEXT
130 :
140 REM************************
150 REM**  S E R P E N T E   **
160 REM**                    **
170 REM** DI GIOVANNI BELLU' **
180 REM**                    **
190 REM**   VIA GIARDINI 20  **
200 REM**                    **
210 REM**    S E R E G N O   **
220 REM**      (MILANO)      **
230 REM**  TEL. 0362-239580  **
240 REM************************
250 REM**  PER COMMODORE 64  **
255 REM**UTILIZZARE  JOYSTICK**
260 REM**  DA INSERIRE NELLA **
270 REM**   CONTROL PORT 1 - **
280 REM************************
290 :
300 POKE54278,248:POKE54276,17:A$="■■■■■■■■":PRINTA$TAB(15)"ECCO A VOI"
:A$=A$+"■"
```

```
310 KJ=56321:FORK=0TO8
320 PRINTA$TAB(15+K)"."
330 A=RND(1)*150:FORJ=1TOA:NEXT:NEXT
340 PRINTA$TAB(15)"          "
350 PRINTRIGHT$(A$,7)TAB(8)"PREMI 'S' PER COMINCIARE":A$=A$+"■"
360 PRINTA$TAB(15)"■ SERPENTE:"
370 PRINTA$TAB(15)"■■◆◆◆◆◆◆◆◆◆◆◆◆"
380 PRINTA$TAB(15)"■■■◆◆          ◆◆"
390 PRINTA$TAB(15)"■■■■◆◆          ◆◆"
400 PRINTA$TAB(15)"■■■■■◆◆          ◆◆"
410 PRINTA$TAB(15)"■■■■■■◆◆◆◆◆◆◆◆◆◆◆◆■":A$=A$+"■■■"
420 POKEQ,RND(1)*255:FORK=1TO20:GETA1$:IFA1$<>"S"THENNEXT:GOTO440
430 GOTO480
440 POKEQ,0:PRINTA$TAB(18)"OOO*"
450 POKEQ,RND(1)*256:FORK=1TO20:GETA1$:IFA1$<>"S"THENNEXT:GOTO470
460 GOTO480
470 POKEQ,0:PRINTA$TAB(18)"■OOO*":GOTO420
480 PRINTA$TAB(18)"OOO*":POKEQ,0
490 PRINT"■■■                                        ": REM 40 SPAZI
500 POKE53281,6:POKE53280,6:INPUT"COME TI CHIAMI";N$
510 PRINT"■":POKE53281,9:POKE53280,0:NM=0:TI$="000000":PT=0:NP=0
511 FORI=55296TO55296+1024:POKEI,7:NEXT
520 FORK=1024TO1064:POKEK,173:NEXT:FORK=1024TO2023STEP40
530 POKEK,173:POKEK+39,173:NEXT
540 FORK=1984TO2023:POKEK,173:NEXT
550 FORK=1043TO2000STEP40:POKEK,173:NEXT
560 PT=0:TI$="000000":NM=0:GOTO650
570 PT=PT+NM*(500-VAL(T$))
580 A1$="■■■■":N=3:PRINTA1$TAB(32)"■000000":A1$=A1$+"■"
590 PRINTA1$TAB(35)"     "
600 PRINTA1$TAB(35)"   ":A1$=A1$+"■■"
610 PRINTA1$TAB(27)"             "
620 TI$="000000":NM=0:PT=PT+5000
630 NR=NR+5:Z=Z+1:IFNP=4THENNP=0::Z=2:NR=15:GOSUB1800
640 IFNP/2<>INT(NP/2)THENGOTO1530
650 GOSUB1730:PRINT"■■"TAB(14)"■-"
655 REM REGOLO "_-_-_-_-_-/_-_-_-_-_-/_-_-_-_-_-/_-_-_-_-_-"
660 PRINT"■■■■■■-----■ ■-----■ ■-■ ■-----■ ■--"
670 PRINT"■■■-■              ■-■ ■-■        ■-"
680 PRINT"■■■-■ ■-----■ ■---■ ■-■ ■-■ ■----■ ■-"
690 PRINT"■■■-■              ■-■ ■-■        ■-"
```

```
700 PRINT"■■■----■ ■--■ ■-----■ ■-■ ■--■ ■---"
710 PRINT"■■              ■-■   ■-■          ■-"
720 PRINT"■■--■ ■-------■ ■-■ ■--------■ ■-"
730 PRINT"■■   ■-■   ■-■  ■-"
740 PRINT"■■■--■ ■-■ ■-■ ■-■ ■-----■ ■-■ ■------"
750 PRINT"■■■-■  ■-■ ■-■ ■-■         ■-■        ■-"
760 PRINT"■■■-■ ■--■ ■-■ ■-■ ■-------■ ■----"
770 PRINT"■■      ■-■   ■-■              ■---"
780 PRINT"■■--------■ ■--■ ■-■ ■-■ ■-■ ■-"
790 PRINT"■■        ■-■     ■-■ ■-■ ■-■ ■-■ ■----"
800 PRINT"■■■-■ ■-■ ■-■ ■-■ ■--■ ■-■ ■-■ ■-■ ■-■ ■-"
810 PRINT"■■■-■ ■-■ ■-■ ■-■     ■-■ ■-■ ■-■ ■-■ ■-■ ■-"
820 PRINT"■■■-■ ■-■ ■-■ ■-■ ■--■ ■-■     ■-■      ■-"
830 PRINT"■■  ■-■      ■-■     ■---------"
840 PRINT"■■■-■ ■-■ ■-----■ ■--■ ■-■     ■-■    ■-"
850 PRINT"■■                   ■-■    ■-"
860 PRINT"■■■----------------------"
870 PRINT"■■■"TAB(27)LEFT$(N$,11)
880 A$="■■■■■■■■■■■■"
890 PRINTA$TAB(30)"■■ DI  "
900 PRINTA$TAB(27)"■■ GIOVANNI  ":PRINTA$TAB(28)"■■■ BELLU'  ■"
910 FORK=1TONR
920 X=INT(RND(1)*25+1):IFX<20ORX>25THEN920
930 Y=INT(RND(1)*25+1):IFY<10ORY>22THEN930
940 IFPEEK(1024+X+40*Y)<>32THEN920
950 POKE1024+X+40*Y,90:POKE55296+X+40*Y,7
960 POKEQ,RND(1)*150+100:FORJ=1TO50:NEXT:POKEQ,0
970 NEXT
980 T=42:C=87:DEFFNA(A)=RND(1)*255
990 A(296)=2024-70:A(297)=A(296)-1:A(298)=A(297)-1:A(299)
=A(298)-1:POKEA(296),T
1000 PRINT"■":N=3:TI$="000000":PRINT"■■■■"TAB(26)"■TIME"
1010 PRINT"■■■■■"TAB(26)"■EATEN■:       "TAB(35)0
1020 D=1:O=296
1030 GOTO1080
1040 A(O+N+1)=A(O+N)
1050 IFTI$>"001000"THENPRINT"■■■■"TAB(32)"■TROPPO TEMPO":GOTO1330
1060 PRINT"■■■■"TAB(32)TI$
1070 POKEQ,FNA(A)
1080 O=O-1:IFO=-1THEN1320
1090 A(O)=A(O+1)+D:X=PEEK(A(O))
```

```
1100 POKEQ,0
1110 IFX=90THENN=N+Z:NM=NM+1:GOTO1520
1120 IFX=87THEN1330
1130 IFNM=NRTHENPRINT"SQQQQQQQ"TAB(28)"■HAI VINTO":T$=TI$:GOSUB1510:NP=
NP+1:GOTO 570
1140 IFX<>32THENA(O)=A(O)-D:D=0:POKEA(O),T:GOTO1240
1150 POKEA(O),T:POKEA(O+1),C
1160 POKEA(O+N+1),32
1170 FORK=1TO10:A%=PEEK(KJ):IFA%=255ORA%<245THENNEXT:GOTO1040
1180 IFA%=253THEND=40:K=10:NEXT:GOTO1230
1190 IFA%=247THEND=1:K=10:NEXT:GOTO1230
1200 IFA%=251THEND=-1:K=10:NEXT:GOTO1230
1210 IFA%=254THEND=-40:K=10:NEXT:GOTO1230
1220 NEXT:GOTO1230
1230 GOTO1040
1240 O=O+1
1250 Y=0:Y1=0:Y2=0:Y3=0:Y4=0
1260 IFPEEK(A(O)+1)=32ORPEEK(A(O)+1)=90THENY=Y+1:Y1=1
1270 IFPEEK(A(O)-1)=32ORPEEK(A(O)-1)=90THENY=Y+1:Y2=-1
1280 IFPEEK(A(O)+40)=32ORPEEK(A(O)+40)=90THENY=Y+1:Y3=40
1290 IFPEEK(A(O)-40)=32ORPEEK(A(O)-40)=90THENY=Y+1:Y4=-40
1300 IFY=1THEND=0:D=D+Y1+Y2+Y3+Y4:GOTO1040
1310 GOTO1170
1320 FORK=0TON:A(298-N+K)=A(K):NEXT:O=298-N:GOTO1040
1330 PT=PT+NM*(500-VAL(TI$))
1340 FORK=1TO250:POKEQ,RND(1)*255:NEXT:POKEQ,0
1350 PRINT"SQQQQQQ"TAB(27)"■ HAI PERSO ":GOSUB1510
1360 FORK=1TO15
1370 PRINT"SQQQQQQQQ"TAB(27)"■PUNTI■"INT(PT)
1380 FORJ=1TO50:NEXT
1390 PRINT"SQQQQQQQQQ"TAB(27)"PUNTI■"INT(PT)
1400 FORJ=1TO50:NEXTJ,K
1410 PRINT"SQQQQQQQ"TAB(28)"■ ANCORA ?"
1420 FORK=1TO50:GETA$:IFA$=""THENNEXT:GOTO1450
1430 IFA$<>"N"ANDA$<>"S"THENNEXT:
1440 GOTO1470
1450 PRINT"SQQQQQQQ"TAB(28)" ANCORA ?"
1460 FORK=1TO50:GETA$:IFA$=""THENNEXT:GOTO1410
1470 IFA$="S"THENPRINT"■":Z=2:NR=15:GOTO500
1480 IFA$<>"N"THENNEXT:GOTO1410
1490 PRINT"■CIAO !!!":PRINT"■■0000000*"
```

```
1500 END
1510 FORK=0TOO+N+1:POKEA(K),32:NEXT:FORK=1TO2000:NEXT:RETURN
1520 FORJ=150TO250:POKEQ,J:NEXT:POKEQ,0:PRINT"■■■■■"TAB(35)NM:GOTO1150
1530 GOSUB1730:PRINT"■■■                                    "
1540 FORK=1TO11
1550 PRINT"■ ■-■ ■-■ ■-■ ■-■ ■-■ ■-■ ■-■ ■-■ ■-■ ■-■ ■--"
1560 PRINT"■                                      "
1570 NEXT:GOTO870
1580 PRINT"■"
1590 PRINTTAB(10)"■ ATTENDERE PREGO "
1600 PRINTCHR$(142):POKE52,48:POKE56,48
1610 POKE56334,PEEK(56334)AND254:POKE1,PEEK(1)AND251:FORK=0TO3584
1620 POKE12288+K,PEEK(53248+K):NEXT
1630 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334)OR1
1640 FORK=0TO7:READA:POKE12288+720+K,A:NEXT
1650 FORK=0TO7:READA:POKE12288+636+K,A:NEXT:FORK=0TO7:READA:
POKE12288+336+K,A
1660 NEXT
1670 RESTORE:FORK=0TO15:READA:NEXT
1680 FORK=0TO7:READA:POKE12288+1360+K,255-A:NEXT
1690 POKE53272,28:RETURN
1700 DATA0,32,86,86,74,138,140,0
1710 DATA60,66,129,153,153,129,66,60
1720 DATA60,66,165,129,165,153,66,60
1730 PRINTE$TAB(27)"■ LIVELLO"NP"■ "
1740 PRINTE$TAB(27)"■■NUMERO ◆=";NR
1750 PRINTE$TAB(27)"■■■■VALORE ◆=";Z
1760 PRINT"■■■                    "
1770 FORK=1TO22
1780 PRINT"■                    "
1790 NEXT:RETURN
1800 FORJ=1TO20
1810 PRINT"■■■"TAB(26)"■ COMPLIMENTI "
1820 FORK=1TO50:NEXT
1830 PRINT"■■■"TAB(26)" COMPLIMENTI "
1840 FORK=1TO50:NEXT:NEXT
1850 PRINT"■■■"TAB(26)"                "
1860 RETURN
1870 REM REGOLO "_‾_‾_‾_‾_/_‾_‾_‾_‾_/_‾_‾_‾_‾_/_‾_‾_‾_‾_

READY.
```

CANESTRO

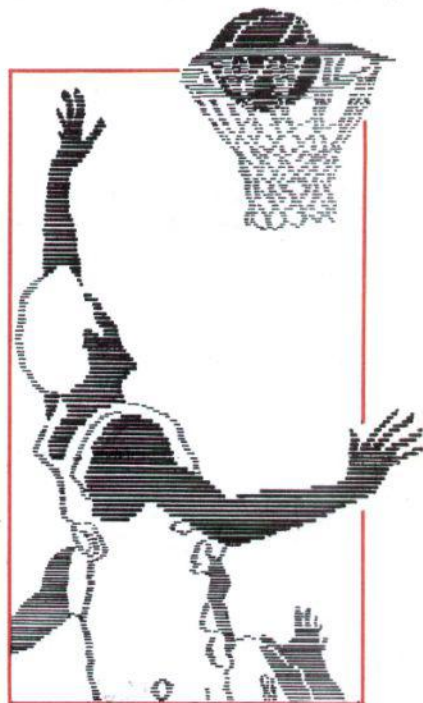
Una pallina un po' dispettosa scende o dalla parte destra o da quella sinistra del video, ed è lei che decide, magari facendovi credere di scendere da una parte mentre poi scende dall'altra, seguendo una traiettoria "quasi" parabolica.

Dico quasi perchè casualmente la pallina potrebbe trovarsi davanti una paletta che la fa rimbalzare, facendole cambiare direzione.

Scopo del gioco è di riuscire a prendere la pallina spostando una specie di canestro che viene disegnato in basso, muovendolo o a destra o a sinistra. In totale scendono 10 palline, e bisogna riuscire a prenderle tutte.

Sulla parte alta del video vengono indicati il numero di palline che sono scese, e quante ne sono state prese.

Il programma è piuttosto breve e chi volesse modificarlo non dovrebbe incontrare difficoltà.

Per esempio si potrebbe modificare il programma in modo da poter giocare in più giocatori contemporaneamente, magari facendo scendere una pallina a turno per ogni giocatore, e alla fine dare una classifica indicando chi è stato il più bravo. ■

```
100 REM"**********************
110 REM"* PALLONE... RANDOM *
120 REM"*  PER COMMODORE 64 *
150 REM"*                   *
160 REM"*     SPOSTAMENTI   *
170 REM"* + = A SINISTRA.   *
180 REM"* - = A DESTRA.     *
190 REM"*                   *
200 REM"**********************
210 POKE54296,15:POKE54277,0:
    POKE54278,248:POKE54276,17
220 DEFFNA(X)=RND(1)*150
230 A$=" XXX ":POKE53280,6
240 B$="■":FORK=1TO23:B$=B$+"■":NEXT
250 X=17:P=197:PRINT"■"
260 REM ************************
270 REM **                    **
280 REM ** DI GIOVANNI BELLU' **
290 REM **                    **
300 REM **   S E R E G N O    **
310 REM **      (MILANO)      **
320 REM **                    **
330 REM ** VIA GIARDINI N.20  **
340 REM **                    **
350 REM **  TEL.0362/239580   **
360 REM **                    **
370 REM ************************
```

```
380 PRINT"[illegible]":Q=54273:POKE53281,0
390 POKE59467,16:POKE59466,16
400 A=1024:FORK=0TO39:POKEA+K,91:POKEA+999-K,102:NEXT
410 FORK=0TO39:POKE55296+K,5:NEXT
420 FORK=ATOA+959STEP40:POKEK,102:POKEK+39,102:NEXT:A=A+200
430 FORK=55296TO56295STEP40:POKEK,5:POKEK+39,5:NEXT
440 FORK=56256TO56295:POKEK,5:NEXT
450 PRINT"[illegible]______"TAB(34)"_____"
460 PRINT"[illegible]‾‾‾‾‾‾"TAB(34)"‾‾‾‾‾"
470 FORK=1TO5:POKEA+K,81:POKEQ,150:GOSUB690:FORJ=1TOFNA(1):
    NEXT:POKEA+K,32
480 POKEQ,0:NEXT
490 IFRND(1)<.3THEN470
500 IFRND(1)<.4THEN520
510 W=A+6:S=1:GOTO570
520 FORK=1TO5:POKEA+39-K,81:POKEQ,150:GOSUB690:FORJ=1TOFNA(1):NEXT
530 POKEA+39-K,32:POKEQ,0:NEXT
540 IFRND(1)<.3THEN520
550 IFRND(1)<.4THEN470
560 W=A+33:S=-1
570 PL=PL+1:PRINT"[illegible]"TAB(17)PL;PT"[illegible]":IFPL=11THEN720
580 PE=PEEK(W+40*Y)
590 IFPE=86THENGOTO710
600 POKEW+40*Y,81:POKEQ,FNA(1)
610 GOSUB690
620 POKEW+40*Y,PE:POKEQ,0
630 IFY>9ANDY<13ANDRND(1)>.8THENPE=PEEK(W+40*Y+S):GOSUB780:GOTO580
640 IFRND(1)<.5THENW=W+S:GOTO580
650 Y=Y+2:IFY<19THEN580
660 IFY>19THENOP=102:GOTO680
670 OP=32
680 POKEW+40*(Y-1),OP:Y=0:W=0:S=0:GOSUB800:GOTO470
690 X1=X:X=X-(PEEK(P)=43)+(PEEK(P)=40):IFX<1ORX>34THENX=X1
700 PRINTB$TAB(X)A$:RETURN
710 FORL=0TO255:POKEQ,L:NEXT:POKEQ,0:PT=PT+1:PRINT"[illegible]"TAB(20)PT"
    [illegible]":Y=0:GOTO520
720 PRINT"[illegible] HAI PRESO "PT" PALLE SU 10"
730 FORK=1TO20:GETA$:NEXT
740 PRINT"[illegible] GIOCHI ANCORA (S/N) ?"
750 GETA$:IFA$<>"S"ANDA$<>"N"THEN750
760 IFA$="S"THENRUN
770 END:END
780 POKEW+40*Y+S,93:W=W-5*S:POKEW+40*Y,81:POKEW+40*Y+6*S,PE:
    POKEW+40*Y,32:S=-S
790 FORKK=100TO250STEP4:POKEQ,KK:NEXT:POKEQ,0:RETURN
800 FORKK=1TO50:POKEQ,FNA(1):NEXT:POKEQ,0:RETURN
810 REM"_‾_‾_‾_‾/_‾_‾_‾_‾/_‾_‾_‾_‾/_‾_‾_‾_

READY.
```

TECNICHE DI OVERLAY

Avvertenza: per seguire il presente articolo, il cui contenuto si rivelerà utilissimo per chiunque possegga un apparecchio Commodore, consigliamo di leggerlo avendo vicino il calcolatore, in modo da seguire *alla lettera* le operazioni da compiere.

E'opportuno altresì che il lettore sappia in che modo è organizzata la memoria RAM del proprio sistema, almeno per ciò che riguarda l'allocazione dei programmi basic e delle variabili (numeriche e stringa). A tale scopo si consiglia di approfondire gli argomenti, sommariamente accennati, nel riquadro al lato.

L'esigenza dell'overlay

A volte capita (e chi possiede un VIC 20 inespanso lo sa fin troppo bene!) che alcuni programmi non possono essere ospitati in un computer perchè sono più lunghi della memoria disponibile. Se infatti si cerca di digitarli una linea alla volta, queste vengono accettate fino a che è possibile. Tentando di proseguire si ottiene null'altro che un messaggio di:

OUT OF MEMORY ERROR

che impedisce il proseguimento della digitazione.

Analogamente programmi anche molto brevi, e di conseguenza "caricabili", contengono alcune istruzioni del tipo DIM che, per essere eseguite, richiedono una certa quantità di memoria.

Sembrerebbe che nei casi appena visti sia impossibile utilizzare i programmi stessi avendo a disposizione una quantità modesta di RAM.

Quasi sempre, però, un certo numero di linee di programma vengono usate per "inizializzare" il programma stesso e non occorrono più nel resto dell'elaborazione. Sarebbe bello se fosse possibile caricare tale parte di programma, farla girare, ed in seguito caricare ed utilizzare la seconda parte. Se però si agisce come è stato descritto, al momento di impartire il RUN, dopo il caricamento della seconda parte, si azzerano tutti i valori delle variabili inizializzate precedentemente.

Per non perdere il contenuto delle variabili, la Commodore specifica che se alla fine della prima parte si inserisce, come ultima riga del programma, una riga del tipo:

1560 LOAD "SECONDA PARTE", 8

il programma con tale nome viene caricato e "lanciato" senza alterare in nessun modo le variabili elaborate dalla prima parte del programma stesso. E' indispensabile, però, che il programma successivo sia più breve di quello che lo "chiama"; vedremo tra breve il perchè.

Purtroppo, anche se in effetti quanto asserito è vero, non sembra esserlo se si cerca di applicare la semplice regoletta con troppa disinvoltura.

Allo scopo di capire in quali casi la procedura può essere applicata e in quali, al contrario, può dar luogo ad inconvenienti di vario tipo, consigliamo di seguire *alla lettera* le fasi indicate. Al termine dell'articolo il lettore sarà sorpreso della semplicità con cui è possibile aumentare le potenzialità di un personal computer.

Primo esperimento

Fase 1). Accendete il computer

oppure, se è acceso, spegnetelo e riaccendetelo in modo da esser sicuri della sua corretta inizializzazione.

Fase 2) Digitate il programma di figura 1 così come è pubblicato, tranne la riga 240. Controllate che funzioni correttamente, digitate la riga 240 e registratelo su di un disco oppure su di un nastro (che numerate con 1). E' ovvio che la sintassi riportata è valida per chi utilizza l'unità a dischi. Chi si serve del registratore a cassette trascriverà semplicemente:
240 LOAD "PROVA N. 1/B"

Tale avvertenza (eliminazione di ",8") vale per tutti i brevi listati presentati nel caso si desideri ricorrere al registratore.

Fase 3) Spegnete e riaccendete il computer. Digitate il programma di figura 2, provatelo (ottenendo una serie di zeri), e registratelo col nome "PROVA N. 1/B) nel modo consueto sullo stesso disco oppure su un altro nastro (N. 2). Utilizzando due nastri si semplificheranno le varie operazioni che stiamo per descrivere.

Fase 4) Caricate, da nastro o disco, il programma **PROVA N. 1/A** e digitate **RUN**. Ciò che accade è piuttosto semplice. Dapprima verrà elaborato il primo programma; in seguito (riga 240) verrà caricato il listato **PROVA N. 1/B**. La visualizzazione delle stesse variabili di prima dimostra

```
160 REM COMMODORE COMPUT.CLUB

READY.
```

Fig. 2/a

che nonostante sia stato caricato un nuovo programma le variabili dichiarate precedentemente non vengono in alcun modo alterate. Se infatti, all'apparizione del consueto **READY** chiediamo il listato, appare null'altro che quello di figura 2, vale a dire quello caricato e lanciato grazie alla riga 240. Sembrerebbe che la Commodore abbia proprio ragione. Ma... è davvero tutto in ordine?

Fase 5) A questo punto, (vale a dire avendo in memoria il programma **PROVA N. 1/B** caricato dal programma **PROVA N. 1/A**), aggiungiamo una linea qualunque come la:
160 REM COMMODORE COMPUT.CLUB (fig. 2a)

Se ora chiediamo il listato, esso ci apparirà più lungo di una sola riga e comunque più breve di quello di figura 1.

Fase 6) Registriamo il programma così formato, con lo stesso nome di prima in modo che possa esser richiamato nuovamente dal programma di figura 1.

Il modo di effettuare questa operazione dovrebbe esser noto ma lo riassumiamo brevemente:
SAVE "@0: PROVA N. 1/B",8
nel caso si possegga un drive 1540 o 1541, e:
SAVE "PROVA N. 1/B"
utilizzando il nastro 2 (vale a dire cancellando il precedente pro-

```
100 REM PROVA N.1/A
110 :
120 PRINT "■INIZIO PROVA 1/A"
130 A=1:B=2:C=3:D=4
140 E=1.2: F=2.3
150 G=3.4: H=5.6
160 PRINT A;B;C;D;
170 PRINT E;F;G;H
180 PRINT "■FINE PROVA 1/A"
190 :
200 REM QUESTE TRE RIGHE
210 REM SERVONO SOLO PER
220 REM ALLUNGARE IL BRODO
230 :
240 LOAD "PROVA N.1/B",8

READY.
```

Fig. 1

Puntatori del basic

1/ L'area di memoria disponibile (RAM) a disposizione dell'utente è formata da una certa quantità di byte, posti di seguito, il cui numero dipende dal computer adoperato. Negli esempi che seguono ci riferiamo al Vic 20 inespanso e al Commodore 64.

2/ Nel Vic 20 la prima locazione libera è la 4096. Nel 64 è invece la 2048. Nel seguito del testo esse verranno indicate con LI (Locazione - Inizio).

3/ L'ultima locazione occupata da un programma Basic varia, come è intuitivo, a seconda della lunghezza del programma stesso. Essa verrà indicata con LF (Locazione-Fine).

4/ Il cosiddetto "indirizzo" di partenza del programma basic viene calcolato esaminado il contenuto di due locazioni di memoria, identiche nel caso del Vic e del 64. Esse sono la 43 e la 44.

5/ Per conoscere, dunque, l'indirizzo di partenza del programma sarà sufficiente eseguire il calcolo:

LI=PEEK(43)+PEEK(44)*256

In genere il risultato porterà ai valori prima visti, vale a dire 4096 nel caso del Vic inespanso e 2048 nel 64.

6/ Altre due locazioni (la 45 e la 46) contengono, in "codice", l'indirizzo dell'ultima locazione interessata dal programma basic presente in memoria in quel momento:

LF=PEEK(45)+PEEK(46)*256

LF, come già detto, cambia a seconda della lunghezza del programma presente.

7/ Non appena si accende l'apparecchio, LI coincide con LF.

8/ Col termine "*puntatore*" si intende il valore numerico della locazione interessata in un'operazione. Ad esempio le locazioni 43 e 44 "puntano" all'inizio del programma basic, mentre le 45 e 46 puntano alla sua fine.

Gestione delle variabili stringa

I computer Commodore allocano le variabili stringa in due zone della memoria. Una di queste si trova *al di là* dell'indirizzo di fine basic. Alcune variabili, invece, vengono individuate, dal Sistema Operativo, *all'interno* del programma basic stesso!

Tale metodo di memorizzazione dei dati (tipico delle sole variabili stringa), che tra breve verrà in parte descritto, ha lo scopo di ottimizzare l'occupazione della memoria RAM. Vediamo come:

Se in un programma basic figura una linea del tipo:

100 A$= "ABCDE": PRINT A$

il Sistema operativo non ha motivo di depositare i caratteri alfanumerici "ABCDE" in una zona di memoria esterna a quella occupata dal programma basic. Poichè infatti tali caratteri si trovano di già in una zona RAM, a quale scopo depositarli da qualche altra parte creando un inutile doppione? In altre parole quando il Sistema Operativo incontra un'istruzione del tipo PRINTA$, andrà a rintracciare l'area della memoria alla quale sono associati i vari caratteri. Nel caso del microprogramma appena visto, questa si trova esattamente ad otto byte di distanza dall'inizio del programma stesso, vale a dire: 2 byte di link, 2 di numerazione basic, 2 del nome della variabile, 1 per il simbolo (=) ed uno, infine, per il simbolo degli apici ("). Un ragionamento del tutto simile viene seguito nel caso di:

READ A$:... DATA "ABCDE"

Ben diverso è il caso della linea basic:

100 A$="ABC"+"DEF": PRINT A$

In questo caso (come in tutti i casi di manipolazione di stringhe) il Sistema Operativo *deve* allocare la stringa A$ (ottenuta come elaborazione di due gruppi di caratteri) in qualche parte della memoria, *diversa* da quella relativa ai prossimi basic. Una cosa è infatti disporre dell'indirizzo di partenza in cui si trovano *tutti* i caratteri relativi alla stringa, altra cosa è invece disporre dell'indirizzo in cui è situata un'operazione di concatenazione di stringhe. Lo scopo, come già detto, è quello di ottimizzare l'occupazione di memoria. Per sincerarvene, digitate e fate girare i due micro programmi seguenti:

```
100 PRINT FRE(0)
110 A$="ABCDEFGHIJ": PRINT A$:
PRINT FRE(0)
```

... e questo ...

```
100 PRINT FRE (0)
110 A$= "ABCDE" + "FGHIJ":
PRINT A$: PRINT FRE (0)
```

Quest'ultimo sembra essere più lungo del precedente di appena tre byte ("+"). Infatti è proprio così e ce lo dimostra il valore di FRE(0) che appare prima della visualizzazione di ABCDEFGHIJ. Il secondo valore che appare, indica, invece, il numero di byte a disposizione *dopo* che la variabile A$ è stata dichiarata. Mentre nel primo programma la variabile A$ è all'interno del programma stesso, nel caso successivo A$ costituisce il risultato di una manipolazione e viene pertanto situata automaticamente in altra zona occupando, di conseguenza una quantità di memoria maggiore, rispetto al precedente macroprogramma, del numero di caratteri costituenti la stringa somma.

Conclusioni

1) Se in un programma viene definita una variabile stringa, essa non occupa altra zona se non quella all'interno del programma basic stesso e viene individuata da puntatori che, nonostante siano situati, come tutti i puntatori, *al di fuori dell'area destinata al basic,* puntano a quelle locazioni di memoria.

2) Se la variabile stringa è il frutto invece di una manipolazione (RIGHT$ LEFT$ ecc.) di una somma (A$+B$), essa viene allocata, come i suoi puntatori, in una zona situata al di fuori dell'area riservata ai programmi basic.

Una descrizione dettagliata di come il Sistema Operativo gestisce le stringhe ed i suoi puntatori esula dallo scopo del presente articolo.

```
100 REM PROVA N.1/B
110 PRINT
120 PRINT "■INIZIO PROVA 1/B"
130 PRINT A;B;C;D;
140 PRINT E;F;G;H
150 PRINT "■FINE PROVA 1/B"

READY.
```

Fig. 2

gramma ivi registrato) nel caso si usi il registratore.

Fase 7) Spegnete, riaccendete e caricate da nastro o disco il "vecchio" programma di figura 1 (PROVA N. 1/A).

Impartite il RUN. Le istruzioni del primo programma, come è facile verificare, vengono correttamente eseguite. Non appena, però, viene caricato e lanciato il secondo programma ci accorgiamo che il risultato è diverso da quello visto nella fase 4. Ciò è accaduto, come vedremo tra breve, perchè il secondo programma è più lungo del primo ! Come è possibile che ciò sia avvenuto? Chiediamo il listato (LIST): esso è proprio identico a quello di figura 2 con in più la sola riga aggiunta nella fase 5: è comunque ben più breve del primo!

Ebbene il secondo programma *sembra* più breve del primo, ma risulta *effettivamente* più lungo. Vediamo di spiegarcene il perchè.

Torniamo ad esaminare la fase 4 e confrontiamola con quanto segue. Quando il programma PROVA N. 1/A, giunto alla riga 240, carica il programma PROVA N. 1/B, questo viene allocato a partire da LI. I puntatori di LF rimangono però *inalterati*. Ciò è dovuto al fatto che le variabili precedentemente dichiarate iniziano a partire da LF. Dunque se un programma, ad un certo punto di un'elaborazione, "chiama" un altro programma, questo, dopo il caricamento, viene considerato lungo quanto il precedente anche se è composto da una sola riga! . Pertanto il sistema operativo del computer "vede", dopo il caricamento di un programma, un listato che termina in LF. Se a questo punto aggiungiamo una riga, questa viene allocata come di consueto, ed i puntatori di fine basic, di conseguenza, alterati come se fosse stata aggiunta ad un programma più lungo. Quanto descritto è proprio l'errore che (intenzionalmente) vi abbiamo fatto commettere aggiungendo la riga 160 nella fase 5. Questa è stata infatti aggiunta *dopo* che il programma di figura 2 era stato caricato automaticamente da PROVA N. 1/A. Ciò dimostra che effettivamente, nel caso di ricorso a tecniche overlay, è indispensabile che il primo programma da caricare in memoria sia il più lungo di quello o quelli che in seguito saranno richiamati in cascata. La spiegazione dell'inconveniente è piuttosto semplice. Dopo l'area basic inizia l'area del-

```
100 REM PROVA N.2/A
110 :
120 PRINT "■INIZIO PROVA 2/A"
130 A=1:B=2:C=3:D=4
140 E=1.2: F=2.3
150 G=3.4: H=5.6
160 PRINT A;B;C;D;
170 PRINT E;F;G;H
180 PRINT "INIZIO MEMORIA =";
190 PRINT PEEK(43)+PEEK(44)*256
200 PRINT "FINE MEMORIA =";
210 PRINT PEEK(45)+PEEK(46)*256
220 PRINT "■FINE PROVA N.2/A"
230 :
240 REM QUESTE TRE RIGHE
250 REM SERVONO SOLO PER
260 REM ALLUNGARE IL BRODO
270 :
280 LOAD "PROVA N.2/B",8

READY.
```

Fig. 3

```
100 REM PROVA N.2/B
110 :
120 PRINT "▮INIZIO PROVA 2/B"
130 PRINT A;B;C;D;
140 PRINT E;F;G;H
150 PRINT PEEK(43)+PEEK(44)*256
160 PRINT PEEK(45)+PEEK(46)*256
170 PRINT "▮FINE PROVA 2/B"

READY.
```

Fig. 4

```
180 REM COMMODORE COMPUT.CLUB

READY.
```

Fig. 4/a

le variabili. Se "invadiamo" una zona di quest'ultima con un programma, la "coda" di questo ne cancella alcune ed esattamente quelle che per prime erano state dichiarate nel corso dell'elaborazione del primo programma.

Verifica del primo esperimento

1/ Per verificare quanto è stato affermato, digitare il programma PROVA N. 2/A (figura 3) senza la linea 280; dopo aver verificato che funziona correttamente, digitare la riga 280 e registrarlo col nome PROVA N. 2/A.

2/ Per sicurezza spegnete, riaccendete e digitate il programma PROVA N. 2/B (figura 4). Una volta controllatone il corretto funzionamento, trascrivete su di un foglio di carta i valori forniti dalle righe 150 e 160. Registratelo dunque col nome PROVA N. 2/B in modo che in seguito possa esser richiamato dal programma di figura 3.

3/ Carichiamo PROVA N. 2/A e "lanciamolo". Al termine dell'elaborazione (dopo cioè che viene caricato e lanciato PROVA N. 2/B riga 280), ci accorgiamo che il secondo programma, benchè più breve del primo, possiede gli stessi puntatori di fine basic. In ogni caso i puntatori indicano valori maggiori di quelli precedentemente trascritti sul foglio di carta.

4/ Col programma PROVA N. 2/B (che è ora allocato in memoria perchè richiamato da PROVA N. 2/A), aggiungiamo la riga 180 di figura 2b oppure una qualsiasi altra riga.

5/ Digitiamo RUN e Return. I valori delle variabili (A,B,C,D,E,F) sono ovviamente tutti nulli non solo perchè è stato dato il RUN, ma anche perchè è stata aggiunta una riga. Ricordiamo, per inciso, che l'inserimento, la cancellazione o la semplice modifica di una riga del programma azzera sempre e comunque qualsiasi variabile memorizzata.

Possiamo dunque notare che i puntatori di fine basic (riga 160) vengono incrementati.

Conclusioni sul primo esperimento

1/ In un programma è possibile, mantenendo inalterate le variabili numeriche, dare l'ordine di caricare un altro programma, purchè quest'ultimo sia più breve o al massimo di eguale lunghezza di quello "chiamante".

2/ Il programma chiamato non deve in nessun caso essere manipolato, pena la perdita della caratteristica di "brevità".

3/ Nel caso si desideri modificare il programma chiamato, procedere come segue:

- spegnere e riaccendere il computer;
- caricare il programma che si desidera modificare;
- modificarlo e registrarlo nuovamente con lo stesso nome di prima avendo l'accortezza di verificare che le modifiche apportate non l'abbiano reso più lungo del programma che in seguito lo chiamerà. Per conoscere la quantità di byte occupata da un programma, digitare NON il semplice PRINT FRE(0) ma:

CLR: RESTORE: PRINT FRE(0) (Return).

Secondo esperimento

Per seguire senza difficoltà quanto segue, si consiglia di comprenderne dapprima il secondo riquadro pubblicato.

Fase 1) Spegnate e riaccendete il computer. Digitate, *così com'è pubblicato* (senza alcuna modifica), il programma di figura 5,

```
100 REM PROVA STRINGHE 1/A
110 A$=" A. DE SIMONE "
120 B$="C.C.CLUB"
130 C$="▣"+A$+B$
140 PRINT A$,B$,C$: PRINT
150 REM RIGA DI RIEMPITIVO
160 REM RIGA DI RIEMPITIVO
170 REM RIGA DI RIEMPITIVO
180 LOAD"STRINGHE N.1/B",8

READY.
```

Fig. 5

(**PROVA STRINGHE 1/A**) privo della riga 180 in modo da evitare un FILE NOT FOUND ERROR. Verificatelo e registratelo dopo aver aggiunto la riga 180.

Fase 2) Spegnete e riaccendete il computer. Digitate e registrate (col nome **STRINGHE N. 1/B**), dopo un opportuno controllo, il listato di figura 6 avendo l'accortezza di trascriverlo senza alcuna modifica. In particolare facciamo notare la *coppia* di doppi punti (::) presenti, dopo i REM, nelle righe 110 e 120.

Fase 3) Spegnete e riaccendete il computer. Caricate e lanciate il programma **PROVA STRINGHE 1/A**... Sorpresa!

Invece di visualizzare, in seguito al caricamento automatico di riga 180, ciò che ingenuamente ci saremmo aspettati (le stringhe **A$** e **B$** di riga 110 e 120 del *primo* programma), compaiono sullo schermo i caratteri delle righe 110 e 120 del *secondo* programma nonostante siano preceduti da due istruzioni REM. Che cosa è successo?

Esaminiamo con attenzione il listato di figura 5. Le righe 110 e 1 2 0 *definiscono* una stringa, mentre la 130 effettua una *concatenazione* tra stringhe. Il Sistema Operativo del calcolatore, di conseguenza, individuerà **A$** e **B$** mediante puntatori che punteranno *all'interno* del programma. Per C$, invece, sia i puntatori che i caratteri costituenti C$, si troveranno al di *fuori* dell'area basic.

Quando la riga 180 carica il secondo listato, *tutti* i puntatori, come già visto nel primo esperimento, *non* vengono modificati e, se nel visualizzare C$ non sorgono problemi, ne sorgono, eccome, nel far apparire **A$** e **B$**. I puntatori di queste variabili, infatti, puntano ancora alle locazioni di inizio stringhe del *primo* programma che, dopo il caricamento del secondo listato, non contengono più i caratteri di riga 110 e 120, ma quelli situati dopo le REM del nuovo programma.

Ecco spiegato il perchè dell'insolita visualizzazione, come pure il motivo dei due caratteri di d o p pio punto situati dopo i REM. In questo modo, i due programmi di figura 5 e 6 risultano (per ciò che riguarda le righe 100, 110 e 120) perfettamente identici agli effetti del funzionamento dei puntatori.

Conclusioni sul secondo esperimento

1) Il programma chiamante non deve contenere istruzioni del tipo:

A$ = "NOME"
oppure
READ A$.... DATA "NOME"

In caso contrario se la variabile stringa viene interessata dal programma chiamato in overlay, nel migliore dei casi compaiono caratteri incomprensibili, nel peggiore possono verificarsi malfunzionamenti di tale entità da essere costretti ad interrompere l'elaborazione.

```
100 REM PROVA STRINGHE 1/B
110 REM::"ATTENZIONE!!!!"
120 REM::"GUARDA!!"
130 PRINT A$,B$,C$

READY.
```

Fig. 6

2) E' pertanto necessario - purtroppo - esaminare riga per riga il programma chiamante e modificare tutte le variabili stringa definite nel modo A$ = "NOME". Un suggerimento: l'istruzione...

A$ = "NOME"

... modificatela in ...

A$ = ""+ "NOME" (vale a dire stringa nulla + "NOME")

... oppure ...

READ A$: A$ = "" + A$: DATA "NOME"

READ A$ (I): A$ (I) = "" + A$(I): DATA "NOME"

Terzo esperimento (o meglio... trucco)

Come caricare dapprima un programma breve e poi uno più lungo?

Alcuni testi suggeriscono di alterare i puntatori di inizio e fine basic prima di effettuare l'operazione che, in effetti, è "proibita", dato che, come abbiamo visto, distrugge parte delle variabili già elaborate.

E' però possibile effettuare l'operazione in modo semplicissimo. Supponiamo di voler utilizzare dapprima il programma di figura 7 che, giunto alla riga 120, riceve l'ordine di caricare il programma "*lungo*" (figura 8) che, come si vede, occupa un numero di byte decisamente superiore.

Ebbene, le fasi da seguire sono le seguenti:

1/ Digitare il programma più breve (fig 7) e registrarlo col nome "BREVE".

2/ Digitare il programma di figura 8 e registrarlo col nome "LUNGO".

3/ A questo punto (avendo cioè "LUNGO" in memoria) digitare RUN 210. Questa operazione caricherà il programma "BREVE" e lo renderà, come abbiamo avuto modo di studiare, di lunghezza eguale a "LUNGO". Dopo l'elaborazione verrà richiamato (riga 120) nuovamente il listato di figura 8 che risulta essere di lunghezza pari (ma guarda un po'...) al programma chiamante!

```
100 REM PROGRAMMA BREVE
110 A=10: B=20: C=30
120 LOAD"LUNGO",8

READY.
```

Fig. 7

Applicazione pratica

Il lettore Marco Navalesi, che possiede un Vic 20 inespanso, ha inviato un programma decisamente interessante sotto molti punti di vista.

Pubblichiamo pertanto il listato originale e quello diviso in due tronconi che si richiamano a vicenda con la tecnica appena descritta.

E' ovvio che bisogna caricare dapprima la parte due e digitare RUN 450. Questa operazione, tra l'altro, dimostra non solo che è possibile aumentare "virtualmente" la memoria del Vic inespanso, ma evidenzia anche il semplicissimo trucco per caricare dapprima un listato breve e poi un programma lungo. ▪

Alessandro De Simone

```
100 REM PROGRAMMA LUNGO
110 :
120 PRINT A: PRINT B: PRINT C
130 :
140 REM QUESTE RIGHE
150 REM SERVONO SOLO PER
160 REM ALLUNGARE IL
170 REM BRODO.
180 :
190 END: REM PENULTIMA RIGA
200 ESEGUIRE : RUN 210
210 LOAD"BREVE",8

READY.
```

Fig. 8

VIC BATTLE

Si hanno due minuti nei quali è possibile distruggere più astronavi nemiche. Il programma gira sul Vic 20 *inespanso* ed è bene non inserire spazi nelle linee perchè il programma occupa tutta la memoria. I comandi sono: "." per far ruotare l'astronave in senso orario, "," in senso antiorario, "/" per sparare, ed i tasti Shift e Commodore per far muovere a varie velocità l'astronave. Questo programma è molto interessante perchè usa tutto lo schermo del televisore, grazie ad una mappa di 28 x 36 (occupa 1008 bytes). Questo avviene

agendo sulle locazioni **36866** e **36867**, facendo partire la **mappa** dello schermo da **7168** invece che **7680**, e **centrando poi** lo schermo sul televisore **modificando** le locazioni **36864** e **36865**. I caratteri definiti **dall'utente sono** posti tra **6144** e **7168** (e **questo** spiega perchè **il programma è così** corto - circa **2K**). **Un'ultima puntualizzazione**: l'istruzione **Peek** (633) nella linea **36** determina se sono stati premuti i tasti **Shift**, Commodore o **Ctrl**. ▪

Marco Navalesi
Via Matteotti, 91 - 54011 Aulla (Massa)
Tel. 0187/402627 prefer. h 18

```
100 REM ***      VIC BATTLE PARTE UNO     ***
110 REM ***                               ***
120 REM ***      BY MARCO NAVALESI        ***
130 REM *** & OVERLAY BY A. DE SIMONE     ***
140 :
150 POKE56,24:POKE52,24:CLR:GOSUB230:POKE36879,8:PRINTCHR$(8):
    A=30720:D(6)=29
160 VS=36877:VI=36878:POKEVI,15:L=0:D(1)=-1:D(2)=-29:D(3)=-28:
    D(4)=-27:D(5)=1
170 D(7)=28:D(8)=27:W(1)=64:W(2)=77:W(3)=93:W(4)=78:W(5)=64:W(6)=77:W(7)=93
180 POKE36866,28:POKE36867,72:POKE36864,7:POKE36865,19:POKE648,28:PRINT"♥"
190 POKE648,30:PRINT"♥":FORI=0TO5:POKE7674+I,32:POKE7674+A+I,1:NEXT:W(8)=78
200 P=7680:R=1:U1=32:U=7800:R1=1:FORI=1TO99:POKE7168+RND(1)*1008,74+RND(1)*2
210 NEXT:P1=32:FORI=1TO15:POKE7168+RND(1)*1008,76:NEXT:TI$="000000"
220 LOAD"BATTLE2",8
230 FORI=0TO1023:POKE6144+I,PEEK(32768+I):
    NEXT:POKE36869,254:FORI=0TO87:READA
240 POKE6672+I,A:NEXT:RETURN
```

```
250 DATA0,7,30,254,30,7,0,0,0,64,48,60,31,30,12,4,
    16,16,16,56,56,124,124,68
260 DATA0,2,12,60,248,120,48,16,0,224,120,127,120,224,0,0,
    16,48,120,248,60,12
270 DATA2,0,68,124,124,56,56,16,16,16,8,12,30,31,60,48,64,0,
    0,0,8,0,0,0,0,0,0
280 DATA0,0,0,64,0,0,0,0,0,0,6,6,0,0,0

READY.
```

```
100 REM ***     VIC BATTLE PARTE DUE       ***
110 REM ***                                ***
120 REM ***      BY MARCO NAVALESI         ***
130 REM ***  & OVERLAY BY A. DE SIMONE     ***
140 :
150 T=PEEK(197):Q=0:IFT=29THENQ=-1:POKEVS-1,225
160 IFT=37THENQ=1:POKEVS-1,227
170 IFT=30THEN320
180 R=R+Q:IFR<1THENR=8
190 POKEVS-1,0:IFR>8THENR=1
200 POKEP,P1:P=P+PEEK(653)*D(R):IFP>8176THENP=P-1008
210 IFP<7168THENP=P+1008
220 P1=PEEK(P):POKEP,65+R:POKEP+A,7:IFP1=65+R1THEN440
230 IFRND(1)>.6THENR1=R1+INT(RND(1)*3)-1:POKEVS,235+RND(1)*18
240 IFR1<1THENR1=8
250 IFR1>8THENR1=1
260 POKEVS,0:POKEU,U1:U=U+D(R1):IFU>8176THENU=U-1008
270 IFU<7168THENU=U+1008
280 U1=PEEK(U):POKEU,65+R1:POKEU+A,5:IFU1=65+RTHEN440
290 IFTI$<"000200"THEN150
300 PRINT"TEMPO SCADUTO";
310 PRINT"PUNTEGGIO:"SC:SC=0:FORI=1TO3000:NEXT:WAIT197,63:GOTO450
320 IFF<>0THEN340
330 Y=D(R):F1=R+65:F=P:L=0:K=R:G=PEEK(36868)OR128
340 POKEVS-1,G:F=F+Y:F1=PEEK(F):POKEF,W(K)
350 IFF<7168ORF>8176THENPOKEF,F1:POKEVS-1,0:GOTO180
360 POKEVS-1,0:IFF1=R1+65THEN430
370 L=L+1:IFL=12THENPOKEF,F1:F=0:GOTO180
380 POKEF,F1:GOTO340
390 POKEU,42:FORI=1TO4:FORJ=1TO8:POKEU+D(J)*I,RND(1)*2+74:
    POKEVS,220+J*3:NEXTJ,I
400 FORI=1TO4:FORJ=1TO8:POKEU+D(J)*I,32:POKEVI,15-I*3:NEXTJ,I
410 POKEVS,0:POKEVI,15:POKEU,32:U=7168+INT(RND(1)*28):U1=32:F=0
420 R1=INT(RND(1)*8+1):RETURN
430 GOSUB390:SC=SC+1:FORI=1TO10:POKE7168+RND(1)
    *1008,RND(1)*2+74:NEXT:GOTO180
440 GOSUB390:PRINT" COLLISIONE";:GOTO310
450 LOAD"BATTLE1",8

READY.
```

VIC 20, ORA

```
0 REM VIC BATTLE-M.NAVALESI
1 POKE56,24:POKE52,24:CLR:GOSUB9000:POKE36879,8:PRINTCHR$(8):A=30720:D(6)=29
2 VS=36877:VI=36878:POKEVI,15:L=0:D(1)=-1:D(2)=-29:D(3)=-28:D(4)=-27:D(5)=1
3 D(7)=28:D(8)=27:WK(1)=64:WK(2)=77:WK(3)=93:WK(4)=78:WK(5)=64:WK(6)=77:WK(7)=93
4 POKE36866,28:POKE36867,72:POKE36864,7:POKE36865,19:POKE648,28:PRINT""
5 POKE648,30:PRINT"":FORI=0TO5:POKE7674+I,32:POKE7674+A+I,1:NEXT:WK(8)=78
6 P=7680:R=1:U1=32:U=7800:R1=1:FORI=1TO99:POKE7168+RND(1)*1008,74+RND(1)*2
7 NEXT:P1=32:FORI=1TO15:POKE7168+RND(1)*1008,76:NEXT:TI$="000000"
30 T=PEEK(197):Q=0:IFT=29THENQ=-1:POKEVS-1,225
32 IFT=37THENQ=1:POKEVS-1,227
33 IFT=30THEN100
34 R=R+Q:IFR<1THENR=8
35 POKEVS-1,0:IFR>8THENR=1
36 POKEP,P1:P=P+PEEK(653)*D(R):IFP>8176THENP=P-1008
37 IFP<7168THENP=P+1008
38 P1=PEEK(P):POKEP,65+R:POKEP+A,7:IFP1=65+R1THEN200
40 IFRND(1)>.6THENR1=R1+INT(RND(1)*3)-1:POKEVS,235+RND(1)*18
42 IFR1<1THENR1=8
43 IFR1>8THENR1=1
44 POKEVS,0:POKEU,U1:U=U+D(R1):IFU>8176THENU=U-1008
45 IFU<7168THENU=U+1008
46 U1=PEEK(U):POKEU,65+R1:POKEU+A,5:IFU1=65+RTHEN200
49 IFTI$<"000200"THEN30
50 PRINT"TEMPO SCADUTO";
51 PRINT"PUNTEGGIO:"SC:SC=0:FORI=1TO3000:NEXT:WAIT197,63:GOTO2
100 IFF<>0THEN110
102 Y=D(R):F1=R+65:F=P:L=0:K=R:G=PEEK(36868)OR128
110 POKEVS-1,G:F=F+Y:F1=PEEK(F):POKEF,WK(K)
111 IFF<7168ORF>8176THENPOKEF,F1:POKEVS-1,0:GOTO34
120 POKEVS-1,0:IFF1=R1+65THEN132
121 L=L+1:IFL=12THENPOKEF,F1:F=0:GOTO34
122 POKEF,F1:GOTO110
129 POKEU,42:FORI=1TO4:FORJ=1TO8:POKEU+D(J)*I,RND(1)*2+74:
    POKEVS,220+J*3:NEXTJ,I
130 FORI=1TO4:FORJ=1TO8:POKEU+D(J)*I,32:POKEVI,15-I*3:NEXTJ,I
131 POKEVS,0:POKEVI,15:POKEU,32:U=7168+INT(RND(1)*28):
    U1=32:F=0:R1=INT(RND(1)*8+1):RETURN

132 GOSUB129:SC=SC+1:FORI=1TO10:POKE7168+RND(1)*1008,RND(1)*2+74:NEXT:GOTO34
200 GOSUB129:PRINT" COLLISIONE";:GOTO51
9000 FORI=0TO1023:POKE6144+I,PEEK(32768+I):NEXT:POKE36869,
     254:FORI=0TO87:READA
9010 POKE6672+I,A:NEXT:RETURN
9020 DATA0,7,30,254,30,7,0,0,0,64,48,60,31,30,12,4,16,
     16,16,56,56,124,124,68
9030 DATA0,2,12,60,248,120,48,16,0,224,120,127,120,224,
     0,0,16,48,120,248,60,12
9040 DATA2,0,68,124,124,56,56,16,16,16,8,12,30,31,60,48,64,
     0,0,0,8,0,0,0,0,0,0
9050 DATA0,0,0,64,0,0,0,0,0,0,6,6,0,0,0

READY.
```

Le memorie di massa e la stampante

Negli ultimi tempi la redazione ha ricevuto numerose telefonate di lettori che richiedevano informazioni sull'uso delle periferiche con particolare riferimento alla stampante ed all'unità di memoria a disco. Nei titoletti di questo articolo troverete le domande più ricorrenti.

Sono qui di seguito riportati alcuni concetti fondamentali spiegati attraverso la comoda forma di risposta a quesiti semplici ma della massima importanza.

Che cosa è un disco?

Un disco (noto meglio come floppy disk) è un supporto magnetico destinato a contenere informazioni "dettategli" dal computer. Nei sistemi CBM considerati in questa sede i dischetti hanno una dimensione da 5 pollici 174. Particolari attenzioni devono essere prestate per la loro conservazione ed uso.

Cosa significa "formattare" un disco?

Quando si acquista un disco nuovo, mai usato, esso assomiglia ad un foglio di carta grande come un lenzuolo nonostante le ridottissime dimensioni esterne. Se del foglio di carta vogliamo realizzare un quaderno, lo soddivideremo in un certo numero di fogli eguali che saranno, in seguito, cuciti con una pinzatrice da ufficio. Se invece il grosso foglio di carta lo vogliamo utilizzare per incartare dei prodotti, lo taglieremo secondo altre misure e non sarà necessario cucire i vari pezzi tra loro. Se vogliamo realizzare... coriandoli, la forma finale del foglio sarà ben diversa da quelle viste. Formattare un disco significa, seguendo l'analogia descritta, dare una certa "forma" al disco in modo tale che possa essere utilizzato convenientemente dal computer cui è destinato. Ogni computer, infatti, o meglio ogni drive, ha un suo modo di suddividere ed organizzare le zone magnetiche del disco. Dischetti formattati con un modello Commodore 1541 possono essere letti da un 1540 o da un 4040, ma non, tanto per fare un esempio, da un 8050, nonostante i computer Commodore "parlino" lo stesso basic (o quasi).

A che cosa serve inizializzare un disco?

Precisiamo, anzitutto, che l'inizializzazione è cosa totalmente diversa dalla formattazione. Quest'ultima va fatta una sola volta e, qualunque sia il computer, *distrugge* qualsiasi dato eventualmente presente sul dischetto.

Quando introduciamo il disco nel drive, evidentemente vuol dire che ce ne serviremo per leggere dei dati o per scriverne altri. Il computer deve sapere, al momento opportuno, il posto preciso in cui leggere le informazioni richieste o scriverne delle altre senza, ovviamente, interferire con quelle già presenti. Riccorendo ad un paragone banale possia-

mo dire che, mediante l'inizializzazione, il computer è in grado di riferire ad un "indice" del disco che assomiglia moltissimo all'indice di un libro qualsiasi. Al momento opportuno, in caso, di richiesta di lettura o scrittura dei dati, il computer saprà comportarsi senza commettere errori di sorta. Ecco perchè deve essere effettuata ogni volta che si cambia disco: il computer deve documentarsi sul nuovo... indice. In alcuni drive (Commodore 8050 e 8250) l'inizializzazione è automatica nel senso che viene effettuata non appena si chiude lo sportellino. In altri casi (1541) deve essere richiesta dall'utente. In genere viene sempre fatta automaticamente al momento del SAVE o del LOAD. Può risultare utilissima quando (caso del 1541) il segnalatore luminoso di errore lampeggia.

In che modo si possono registrare dei dati?

Si presuppone che il lettore sappia usare i semplicissimi comandi di registrazione e lettura programmi riportati sul manuale di istruzione.

Registrare dati numerici o alfanumerici è un compito, per il computer, del tutto analogo a quello relativo ai programmi. Mentre però per questi il comando di Save è inequivocabile perchè si tratta di salvare l'intero programma presente in memoria, nel caso dei dati il computer non può sapere quanti e quali dati vogliamo registrare o leggere. E' pertanto necessario comunicare, in qualche modo, che desideriamo leggere oppure scrivere dati. Per fare ciò è necessario "aprire" un file e, dopo aver compiuta l'operazione, chiuderlo. Secondo una banale analogia è un comportamento analogo a quello che seguiamo nel fare una telefonata: dapprima alziamo la cornetta per stabilire la linea, quindi formiamo il numero ed in seguito, terminata la conversazione, chiudiamo la linea. Malfunzionamenti si verificano nel caso formiamo il numero prima di sentire il segnale di linea libera, oppure se lasciamo sollevata la cornetta (= mancanza di chiusura). Allo stesso modo, dall'altro capo del file, l'interlocutore solleverà la cornetta non per formare il numero, ma in risposta allo squillo della nostra chiamata. Sui manuali di istruzione sono riportate le istruzioni e la corretta sintassi di rispettare per realizzare un trasferimento di dati tra computer ed unità a disco.

Che differenza c'è tra un file sequenziale ed uno relativo di random?

Supponiamo di avere memorizzato su disco una rubrica telefonica e di voler aggiungere, in coda ad essa, un nominativo. Se il file è sequenziale siamo costretti a trasferirlo tutto, compresi i nominativi che non interessano, nella memoria del computer, aggiungervi in coda il nuovo nome e trasferire nuovamente su disco il file incrementato come descritto. Benchè tale operazione sia in gran parte automatica, il tempo richiesto è decisamente proibitivo. Se si dispone di file relativi. invece, è possibile aggiungere, nella parte opportuna del disco, il nominativo senza essere costretti all'operazione descritta. Sembrerebbe allora che i file sequenziali siano in netto svantaggio rispetto ai relative; si tenga comunque presente che i dati sequenziali occupano su disco una quantità di memoria minore. In molti casi, inoltre, il vantaggio di trascrivere il singolo dato in coda agli altri già presenti è dal fatto che, in fase di ricerca di una stringa, tanto per fare un esempio, si è costretti a passare in rassegna tutto il file, relativo o sequenziale che sia, prima di rintracciare quella giusta.

Per leggere dati registrati su disco

Una trattazione approfondita sarebbe in questa sede inopportuna. Purtuttavia, referendosi alle figu-

Figura 1

```
100 A=20: B=30: C=76:                 REM TRE VALORI CASUALI
110 OPEN 2,8,2,"0:TEST,SEQ,WRITE":    REM APERTURA COLLOQUIO CON DRIVE
120 PRINT#2,A: PRINT#2,B: PRINT#2,C:REM SCRITTURA SU DISCO
130 CLOSE2:                           REM CHIUSURA COLLOQUIO
```

re 1 e 2 si indicano qui di seguito alcune regole da rispettare nel caso di semplici operazioni di lettura-scrittura utilizzanti dischi.

1/ Ricordarsi *sempre* di chiudere il file aperto per le operazioni.

2/ I dati da leggere devono essere "simmetrici" rispetto a quelli precedentemente scritti. Per esempio i valori A, B e C scritti in quest'ordine col programma di figura 1 devono *assolutamente* essere letti nello stesso ordine nel caso di lettura (fig. 2).

Figura 2

```
100 OPEN 2,8,2,"0:TEST,SEQ,READ":   REM APERTURA COLLOQUIO CON DRIVE
110 INPUT#2,A: INPUT#2,B: INPUT#2,C:REM LETTURA DA DISCO
120 CLOSE2:                         REM CHIUSURA COLLOQUIO
130 PRINT A, B, C:                  REM VERIFICA
```

Se un file è formato da un certo numero di stringhe alfabetiche e da valori numerici, in fase di successiva lettura bisognerà tener conto di questo e dare il comando INPUT#...A$ (o simile) nel caso di stringhe e INPUT#...A nel caso di valori numerici.

3/ Per i motivi detti prima è praticamente impossibile leggere un qualsiasi file se non si conosce il modo in cui è stato scritto. Non dimentichiamo che un file è un elenco di numeri e caratteri che possono non avere apparentemente un nesso logico tra loro. Ecco perchè programmi che ad un certo punto richiedono l'uso del disco per memorizzare dati *devono* essere usati in unione a programmi "simmetrici" di lettura che tengano conto del modo in cui sono stati scritti i dati stessi. Modificare, in un programma, la parte relativa alla sola lettura o scrittura portano a risultati imprevedibili e, in genere, errati.

Come utilizzare la stampante in modo "diretto", senza cioè ricorrere a programmi di word processing

Anche in questo caso è necessario "aprire" il colloquio. Il lettore dovrebbe aver capito che tutte le volte in cui è necessario comunicare qualcosa ad una periferica, ciò si realizza aprendo dei file e chiudendoli dopo l'operazione.

Figura 3

```
100 OPEN1,4:  REM APERTURA COLLOQUIO CON STAMPANTE
110 INPUT A$: REM RICHIESTA DELLA STRINGA ALFANUMERICA DA STAMPARE
120 IF A$="←" THEN GOTO 140: REM PREMERE FRECCIA PER FINE
130 PRINT#1, A$: GOTO 110:   REM SCRITTURA SU STAMPANTE E RITORNO A 110
140 PRINT#1: CLOSE1:         REM CHIUSURA COLLOQUIO CON STAMPANTE
```

La figura 3 rappresenta il più elementare programma di text editor possibile.

• *Riga 100:* si apre il colloquio con la stampante. Il valore 4 di OPEN1,4 si riferisce, appunto, alla stampante come il valore 8, visto precedentemente, è quello tipico dell'unità a dischetti.

• *Riga 110:* il computer chiede all'utilizzatore la stringa alfanumerica che si desidera stampare.

• *Riga 120:* è questo il modo suggerito per interrompere l'invio dei caratteri alla stampante: battendo, al momento della richiesta di una frase, il tasto della freccia a sinistra e, dopo, il tasto di ritorno, il programma salta alla riga 140 che fa stampare un rigo a vuoto e chiude il file. E' ovvio che il lettore potrà cambiare il carattere suggerito e sostituirlo con un altro qualunque.

• *Riga 130:* sul canale 1, precedentemente aperto (riga 100) verrà inviata la stringa alfanumerica A$ digitata. E' questo il momento in cui la stampante scrive i caratteri. Subito dopo (GOTO 110) il programma torna a chiedere una nuova stringa alfanumerica. Si ricorda, benchè sia noto, che è possibile rispondere all'input con una intera frase lunga decine di caratteri.

Vic Agenda

La grande versatilità del Vic 20 si dimostra in modo efficace anche per chi pensa di sfruttarlo per "usi gestionali" senza pretendere però miracoli. Vic-Agenda è un programma che serve a memorizzare su nastro (più "scomodo", ma sicuramente più popolare e diffuso del floppy) un certo numero di indirizzi richiamabili per ordine, numero di codice ed infine per cognome e nome.

Il programma, da solo, occupa circa 4K (3,5K senza le REM). Un problema sorge però, relativamente alla quantità di indirizzi inseriti nell'agenda e quindi al dimensionamento iniziale dei vettori presenti nel listato. Così come si presenta esso gira sul Vic con un minimo di 3K di espansione (8K, 16K, ecc.).

Il programma è molto semplice e versatile (con piccole modifiche e con un po' di creatività si riesce ad adattarlo anche alle esigenze più varie: archivio riviste, componenti elettronici, cassette di registrazione, dischi ecc.), quindi, è superfluo aggiungere altro, oltre al fatto che esso offre 3 menù generali, fra i quali uno principale, che permettono ogni gestione dell'indirizzo: variazioni, aggiunte di altri indirizzi, cancellazioni e la visualizzazione del "file" riguardante la persona in questione identificata dal nome-cognome, via o piazza, C.A.P., località, provincia, telefono, annotazioni varie. ■

Stefano Donati
Via Sgurgola, 13
00179 Roma - tel. 06/7850466

```
10 REM**************************************
20 REM*                                    *
30 REM*  VIC-AGENDA PER VIC CON ESPANSIONE *
40 REM*                                    *
50 REM* DI STEFANO DONATI VIA SGURGOLA  13 *
60 REM*  TEL 06/7850466 -- 00179  ROMA --  *
70 REM*                                    *
80 REM**************************************
85 :
90 W$="┌──────────────────┐":S$="---------------------":U$="└──────
───┘"
```

```
100 DIMN$(50),V$(50),C$(50),L$(50),P$(50),T$(50),A$(50),F(50):GOTO340
110 INPUTB:IFB<0THEN110
120 RETURN
130 PRINT"◘":IFS<>1THEN200
140 OPEN1,1,1,"AGENDA"
150 Z$=CHR$(13):PRINT#1,K$;Z$;C;Z$;
160 FORX=1TOC
170 PRINT"█STO SCRIVENDO             L'INDIRIZZO N.";X
180 PRINT#1,N$(X);Z$;V$(X);Z$;C$(X);Z$;L$(X);Z$;P$(X);Z$;T$(X);Z$;
A$(X);Z$;F(X);Z$;
190 NEXTX:CLOSE1
200 PRINT"◘":RETURN
210 PRINT"◘":IFS=1THEN290
220 OPEN1,1,0,"AGENDA":INPUT#1,K$,C:PRINT"██";W$:PRINT"SCHEDARIO: ";K$
230 PRINT"INDIRIZZI INSERITI:";C:PRINT"█";U$
240 FORX=1TOC
250 PRINT"█STO LEGGENDO              L'INDIRIZZO N° ";X
260 INPUT#1,N$(X),V$(X),C$(X),L$(X),P$(X),T$(X),A$(X),F(X)
270 NEXTX:X=X-1:CLOSE1
280 S=1
290 RETURN
300 PRINT"◘"
310 PRINT"▯";W$:PRINT"▯I█SCHEDARIO  INDIRIZZI■I"
320 PRINT"▯";U$;"██"
330 RETURN
340 GOSUB300
350 PRINT"█0■ FINE LAVORO███":PRINT"█1■ CREAZIONE SCHEDARIO███"
360 PRINT"█2■ AGGIORNAM. SCHEDARIO█████"
370 GOSUB110
380 IFB>2THEN370
390 IFB=0THEN1490
400 IFZ=1THEN420
410 IFB=1THEN510
420 IFB=1THEN370
430 GOSUB300
440 PRINT"▯█0■QUADRO PRINCIPALE██":PRINT"█1■INSERIM.   INDIRIZZI██"
450 PRINT"█2■VARIAZIONE INDIRIZZI██":PRINT"█3■ELIMINAZ.  INDIRIZZI██"
460 PRINT"█4■VISUALIZZ. INDIRIZZI███"
470 GOSUB110
480 IFB=0THEN340:IFB>4THEN470
490 PRINT"◘"
500 ONBGOTO750,790,1080,1130
510 REM*SUBROUTINE CREAZIONE*
520 PRINT"◘"
530 INPUT"█NOME SCHEDARIO■";K$
540 X=1:F(X)=X
550 F(X)=X
560 PRINT"◘":PRINT"▯";W$:PRINT"▯I█RECORD N°:■ ";X
570 PRINT"█I";U$:PRINT"▯COGNOME E NOME":INPUTN$(X)
580 PRINTS$:PRINT"▯VIA O PIAZZA":INPUTV$(X):IFV$(X)="="THEN570
590 PRINTS$:PRINT"▯C.A.P.":INPUTC$(X):IFC$(X)="="THEN580
```

```
600 PRINTS$:PRINT"LOCALITA' ":INPUTL$(X):IFL$(X)="="THEN590
610 PRINTS$:PRINT"PROVINCIA (XX)":INPUTP$(X):IFP$(X)="="THEN600
620 IFLEN(P$(X))<>2THEN610
630 PRINTS$:PRINT"TELEFONO":INPUTT$(X):IFT$(X)="="THEN610
640 PRINTS$:PRINT"ANNOTAZIONI":INPUTA$(X):IFA$(X)="="THEN630
650 PRINT"":PRINT"";S$:PRINT""
660 PRINTN$(X):PRINTV$(X):PRINTC$(X);" ";L$(X);" ";P$(X):PRINT:PRINT"
TEL.:";T$(X)
670 PRINT"NOTE:";A$(X)
680 PRINT"";S$:PRINT"VA BENE (S/N)";:INPUTJ$:IFJ$<>"S"THEN560
690 PRINT:PRINTS$:PRINT"ALTRI INDIRIZZI";:INPUTJ$:IFJ$<>"S"THEN720
700 X=X+1
710 GOTO550
720 C=X:S=1:T=1:Z=1
730 GOTO340
740 REM*SUBROUTINE INSERIMENTO*
750 GOSUB210
760 T=1
770 GOTO700
780 REM*SUBROUTINE VARIAZIONE
790 GOSUB210
800 PRINT""
810 INPUT"CODICE DA VARIARE:";Y:IFY<=0ORY>CTHEN810
820 PRINT"":PRINT"CODICE";F(Y):PRINTS$
830 PRINT"NOME :";N$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$="S"THEN910
840 PRINT"VIA:";V$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$="S"THEN930
850 PRINT"C.A.P.:";C$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$="S"THEN950
860 PRINT"LOCALITA':";L$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$=
"S"THEN970
870 PRINT"PROVINCIA (XX): ";P$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$=
"S"THEN990
880 PRINT"TELEFONO:";T$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$=
"S"THEN1020
890 PRINT"ANNOTAZIONI:";A$(Y):PRINT"VARIO (S/N)";:INPUTJ$:IFJ$=
"S"THEN1040
900 GOTO1050
910 PRINT:INPUT"NUOVO NOME:            ";N$(Y)
920 GOTO840
930 PRINT:INPUT"NUOVA VIA:             ";V$(Y)
940 GOTO850
950 PRINT:INPUT"NUOVO C.A.P.:          ";C$(Y)
960 GOTO860
970 PRINT:INPUT"NUOVA LOCALITA':       ";L$(Y)
980 GOTO870
990 PRINT:INPUT"NUOVA PROVINCIA(XX):";P$(Y)
1000 IFLEN(P$(Y))<>2THEN990
1010 GOTO880
1020 PRINT:INPUT"NUOVO TELEFONO:       ";T$(Y)
1030 GOTO890
1040 PRINT:INPUT"NUOVE NOTE:           ";A$(Y)
1050 PRINT"";S$:PRINT"ALTRI COD. DA VARIARE (S/N)":INPUTJ$:IFJ$=
```

```
"S"THEN800
1060 T=1:GOTO340
1070 REM*SUBROTINE ELIMINAZIONE*
1080 GOSUB210
1090 INPUT"CODICE DA ELIMINARE";Y:IFY<=0ORY>CTHEN1090
1100 N$(Y)="":V$(Y)="":C$(Y)="":L$(Y)="":P$(Y)="":T$(Y)="":A$(Y)=""
1110 PRINTS$:PRINT"ALTRI D'ELIMINARE(S/N)":INPUTJ$:IFJ$="S"THEN1090
1120 T=1:GOTO340
1130 REM*SUBROUTINE VISUALIZZAZIONE
1140 GOSUB210
1150 PRINT""
1160 PRINT"";W$:PRINT"|VISUALIZZ. INDIRIZZI|"
1170 PRINT"";U$;"":PRINT"0 QUADRÓ PRINCIPALE":PRINT"1 RICERCA
PER CODICE"
1180 PRINT"2 RICERCA PER NOME":PRINT"3 VISUALIZZ. TOTALE"
1190 GOSUB110
1200 IFB>3THEN1190
1210 IFB=0THEN340
1220 ONBGOTO1230,1280,1360
1230 PRINT""
1240 INPUT"CODICE DA VISUALIZZ.";Y:IFY<=0THEN1230
1250 IFY>CTHEN1150
1260 GOSUB1400
1270 GOTO1230
1280 PRINT""
1290 INPUT"NOME DA CERCARE      ";Y$:HH=LEN(Y$)
1300 FORY=1TOC
1310 IFLEFT$(N$(Y),HH)<>Y$THEN1340
1320 GOSUB1400
1330 GOTO1280
1340 NEXTY
1350 GOTO1150
1360 FORY=1TOC
1370 GOSUB1400
1380 NEXTY
1390 GOTO1150
1400 PRINT""
1410 PRINT"CODICE N°";F(Y):PRINTS$
1420 PRINTN$(Y):PRINTV$(Y):PRINTC$(Y);" ";L$(Y);" ";P$(Y):PRINT"
TEL:";T$(Y)
1430 PRINT"NOTE:";A$(Y)
1440 PRINT:PRINTS$:PRINT"PREMI 0 PER USCIRE"
1450 PRINT:PRINT"RETURN PER CONTIN.":PRINT"";
1460 GOSUB110
1470 IFB=0THEN1130
1480 RETURN
1490 IFT=0THEN1510
1500 GOSUB130
1510 PRINT"":END

READY.
```

CALENDARIO PERPETUO

Il programma "calendario perpetuo" è stato preparato sul Vic 20 inespanso. Poichè nelle istruzioni non compaiono POKE, esso gira anche sul Vic 20 con espansione di memoria; per essere usato sul Commodore 64. E' necessario apportare piccole modifiche.

Il programma consente di visualizzare sul monitor TV qualunque mese di qualunque anno compreso fra l'anno zero dell'era volgare e l'anno 3000 D.C.

E' richiesta l'indicazione del mese (in cifre da 1 a 12) e l'anno (Dopo Cristo); ottenuti tali dati, viene visualizzato il mese richiesto, e si richiede contemporaneamente l'input di un altro mese: ciò consente di "sfogliare" il calendario senza dover ridare il "RUN" ogni volta.

Il listato si basa sul "Calendario Perpetuo" preparato e disegnato nel 1978 come rielaborazione originale di vari calendari perpetui esistenti. Per redigere il programma in Basic, sono stati aggiunti a quel calendario, alcuni secoli in modo di arrivare al 3000 (tanto per non dare la sensazione di avere un programma troppo miope).

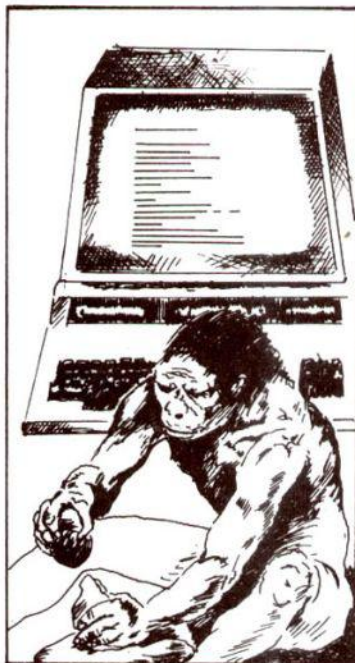



Illustrazione del programma

a/ Da 100 a 170 il programma richiede gli input M (mese) ed A (anno), e controlla ch'essi siano corretti, altrimenti segnala l'errore e ripone la richiesta di input.
b/ Da 180 a 240 si esamina il caso particolare dell'Anno 1582, anno in cui, nel mese di ottobre, si è attuata la riforma del calendario passando dal Calendario Giuliano (che considerava bisestili tutti gli anni secolari) al Calendario Gregoriano (che considera bisestili gli anni secolari di 4 in 4: 1600, 2000, ecc.)

Poichè, per eliminare l'errore accumulatosi dall'epoca di Giulio Cesare al 1582 era necessario eliminare 10 giorni, fu stabilito che il giorno successivo al 4 ottobre fosse denominato 15 ottobre (cioè: i giorni dal 5 al 14 ottobre 1582 non sono in realtà mai esistiti). La subroutine 250-260 realizza questo salto in fase di stampa dei giorni del mese (istruzioni 560-670).
c/ Per tutti gli altri anni (≠ 1582) il programma riprende da 330 ove l'input A viene diviso in due (S= cifre dei secoli; D = cifre delle decadi e degli anni).
d/ Da 340 a 350 si confronta S con le ordinate della tabella 1. I relativi dati sono riportati in 680-690, riga per riga a partire dalla 5^(in cui si ottiene nella 1^ colonna il numero romano I: questo valore si incrementa di 1 ad ogni cambio di ri-

ga). In realtà si inizia ponendo Y=8 (anzichè 8) nell'istruzione 330: ciò equivale ad Y=1 poichè l'istruzione 510 provvederà ad effettuare Y-7 fino ad avere Y<7.
e/ Da 360 a 380 si confronta S con le ascisse della tabella 1. I relativi dati sono riportati in 700-760, colonna per colonna a partire dalla prima. Y=Y+1 ad ogni cambio di colonna.
f/ Da 390 a 400 si stabilisce le durata del mese (F).
g/ Da 410 a 430 si confronta M con la seconda tabella. I relativi dati sono riportati in 770, raggruppando i mesi in modo da iniziare con quelli che hanno un 1 (Luna) sulla 1^ riga, continuando con quelli che hanno 2 (Marte), e così via, in modo da incrementare ancora Y di uno ad ogni cambio di gruppo.
h/ Da 440 a 500 si individuano gli anni bisestili, per i quali si rinvia alla subroutine 270-290 (che riduce Y di uno per GEN e FEB e che pone F=29 per Febbraio).
i/ Da 520 a 540 si traduce M in MS (dizione del mese in tutte lettere prelevata dai dati 780-790) per la successiva stampa.
l/ Da 550 a 670 si provvede alla stampa del mese usando anche la subroutine 300-320 per le linee rosse. Il valore Y stabilisce il giorno della settimana (da LU a DO) in cui inizia il primo giorno del mese; l'istruzione 560 calcola conseguentemente la colonna (X) in cui si andrà a stampare (istruzione 580) il giorno 1 del mese. Gli altri giorni del mese (fino ad F) vengono stampati di seguito, opportunamente incolonnati. I giorni che capitano di Domenica vengono stampati in rosso (istruzioni 620 e 650). ▪

Domenico & Marzio Carro
Via Leonida, 82 C - Taranto
Tel. 099/372358

```
10 REM****************
20 REM  CALENDARIO
30 REM   PERPETUO
40 REM      DI
50 REMDOMENICO&MARZIO
60 REM      CARRO
70 REMVIA LEONIDA,82C
80 REM    TARANTO
90 REM TEL(099)372358
95 REM****************
96 :
100 PRINT"CALENDARIO PERPETUO"
110 PRINT""
120 INPUT"MESE(IN CIFRE)";M
130 IFM=0ORM>12THENPRINT"CIFRE DA 1 A 12":GOTO120
140 PRINT
150 INPUT"ANNO( D.C. )";A
160 IFA<0THENPRINT"HO DETTO DOPO CRISTO":GOTO150
170 IFA>3000THENPRINT"MI DISPIACE:PROGRAMMA  VALIDO FINO AL 3000":GOTO150
180 IFA<>1582THEN330
190 IFM<10THENY=1:GOTO390
200 IFM>10THENY=5:GOTO390
210 IFM=10THENPRINT""SPC(5)"OTTOBRE 1582"
220 PRINTSPC(2)"ANNO DELLA RIFORMA"
230 PRINTSPC(6)"GREGORIANA":GOSUB300
240 Y=1:F=31:GOTO560
250 IFG=5THENG=15
260 RETURN
270 IFM=1ORM=2THENY=Y-1
280 IFM=2THENF=29
290 RETURN
```

```
300 PRINTCHR$(13)
310 FORC=0TO21:PRINTTAB(C)"▚▚ ";:NEXT
320 PRINT"▀▀":RETURN
330 S%=A/100:S=S%:D=A-S*100:Y=8:IFS=15ANDD>82THENS=-15
340 READN:IFN=-1THENY=Y+1
350 IFN<>STHEN340
360 READN:IFN<>-2THEN360
370 READN:IFN=-1THENY=Y+1
380 IFN<>DTHEN370
390 F=31:IFM=2THENF=28
400 IFM=4ORM=6ORM=9ORM=11THENF=30
410 READN:IFN<>-3THEN410
420 READN:IFN=-1THENY=Y+1
430 IFN<>MTHEN420
440 IFD=0THEN480
450 FORC=4TO96STEP4:IFD=CTHENGOSUB270
460 NEXT
470 GOTO510
480 FORC=0TO16:IFS=CTHENGOSUB270
490 NEXT
500 IFS=20ORS=24ORS=28THENGOSUB270
510 IFY>7THENY=Y-7:GOTO510
520 READN:IFN<>-4THEN520
530 FORC=1TO12:READM$
540 IFC<>MTHENNEXT:GOTO530
550 PRINT"▛"SPC(5)M$;" ";A;:GOSUB300
560 X=1+3*(Y-1)
570 PRINT" LU MA ME GI VE SA ▚DO▀ ▚"
580 PRINTSPC(X);
590 FORG=1TOF
600 IFA=1582ANDM=10THENGOSUB250
610 IFG>9THENPRINT"▌";
620 IFPOS(1)>17THEN650
630 PRINTG;:GOTO660
650 PRINT"▚";G;"▚"SPC(1)"▀";
660 IFG=FTHENGOSUB300
670 NEXTG
680 DATA4,11,-15,19,23,27,-1,3,10,-1,2,9,18,22,26,30,-1,1,8,15,-1
690 DATA0,7,14,17,21,25,29,-1,6,13,-1,5,12,16,20,24,28,-2
700 DATA0,6,17,23,28,34,45,51,56,62,73,79,84,90,-1
710 DATA1,7,12,18,29,35,40,46,57,63,68,74,85,91,96,-1
720 DATA2,13,19,24,30,41,47,52,58,69,75,80,86,97,-1
730 DATA3,8,14,25,31,36,42,53,59,64,70,81,87,92,98,-1
740 DATA9,15,20,26,37,43,48,54,65,71,76,82,93,99,-1
750 DATA4,10,21,27,32,38,49,55,60,66,77,83,88,94,-1
760 DATA5,11,16,22,33,39,44,50,61,67,72,78,89,95,-3
770 DATA1,10,-1,5,-1,8,-1,2,3,11,-1,6,-1,9,12,-1,4,7,-4
780 DATAGENNAIO,FEBBRAIO,MARZO,APRILE,MAGGIO,GIUGNO,LUGLIO
790 DATAAGOSTO,SETTEMBRE,OTTOBRE,NOVEMBRE,DICEMBRE
800 RESTORE:GOTO120
READY.
```

DIVISIONE IN SILLABE

Il programmino suddivide in sillabe qualunque parola italiana. L'interesse è più linguistico che informatico, ma può essere utile a molti lettori, grazie anche alla sua brevità.

Per quanto riguarda il funzionamento, è molto semplice: il programma cerca la prima vocale, controlla le due lettere successive e a seconda delle combinazioni individua o meno la sillaba, ricominciando poi da capo il ciclo. L'unica particolarità è la linea 460 che assegna alla variabile **A** il valore -1 solo se una delle condizioni successive è vera, cioè se **X$** è una vocale.

```
100 REM **********************
110 REM  DIVISIONE IN SILLABE.
120 REM PER QUALSISI COMPUTER
130 REM
140 REM      DI FRANCO MUSSO
150 REM       VIA TORINO 27
160 REM    ORIO  CANAVESE  (TO)
170 REM      TEL 011/9838044
180 REM **********************
190 :
200 PRINT"█"
210 S=1:PRINT"█"
220 INPUT"█VOCABOLO█";A$:PRINT
230 IFA$="0"THENEND
240 IFS>LEN(A$)THENPRINT"█ ":GOTO210
250 X$=MID$(A$,S,1):GOSUB460
260 IFA=0THEN430
270 X$=MID$(A$,S+1,1):GOSUB460
280 IFA=0THEN360
290 IFX$="I"THEN320
300 IFMID$(A$,S,1)="I"ORMID$(A$,S,1)="U"THEN430
310 GOTO440
320 X$=MID$(A$,S+2,1):GOSUB460
330 IFS>1THENIFMID$(A$,S-1,2)="QU"ANDA=-1THENPRINTMID$(A$,S,2);:S=S+2:GOTO240
340 IFA=-1THEN440
350 GOTO430
360 IFS+2>LEN(A$)THEN450
370 X$=MID$(A$,S+2,1):GOSUB460
380 IFA=-1THEN440
390 IFMID$(A$,S+1,1)=MID$(A$,S+2,1)THEN450
400 A1$=MID$(A$,S+1,1):IFA1$="S"ORA1$="G"THEN440
410 REM  A2$=MID$(A$,S+2,1):IFA2$="R"ORA2$="L"ORA2$="H"THEN440
420 GOTO450
430 PRINTMID$(A$,S,1);:S=S+1:GOTO240
440 PRINTMID$(A$,S,1);"-";:S=S+1:GOTO240
450 PRINTMID$(A$,S,2);"-";:S=S+2:GOTO240
460 A=X$="A"ORX$="E"ORX$="I"ORX$="O"ORX$="U":RETURN

READY.
```

PUZZLE

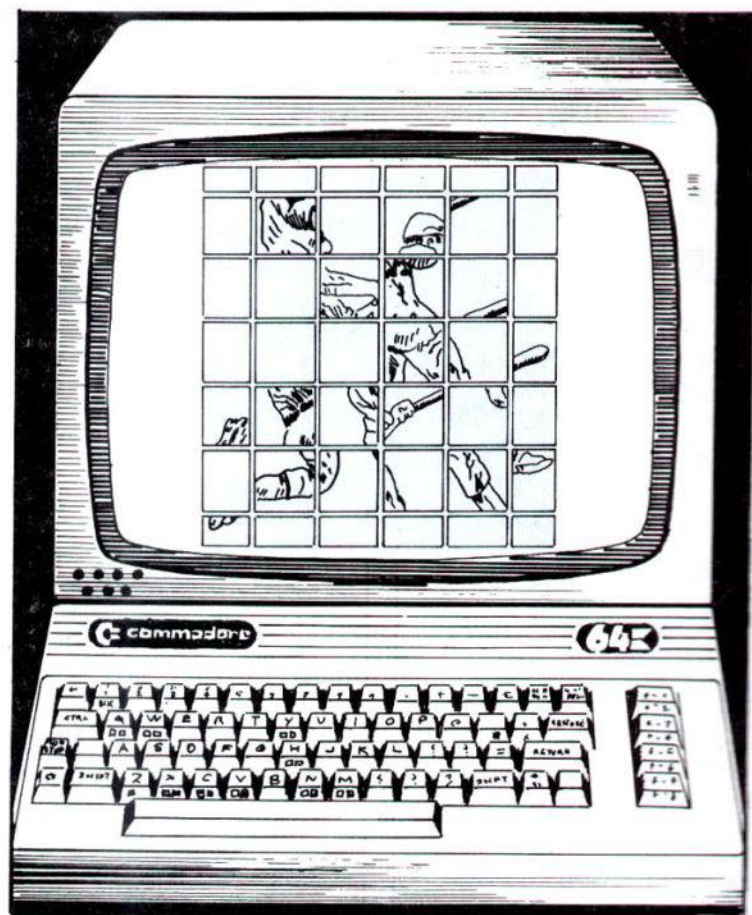
Provate a ricostruire un'immagine avendo a disposizione le tessere-carattere del mosaico.

Appena dato il RUN viene mostrata per alcuni secondi una figura, che poi verrà cancellata e mischiata casualmente dal computer, e per tutta la partita non si può più vedere, a meno che alla domanda "vuoi vedere la figura" si risponda di sì, ed allora sulla destra della figura mischiata apparirà quella "originale".

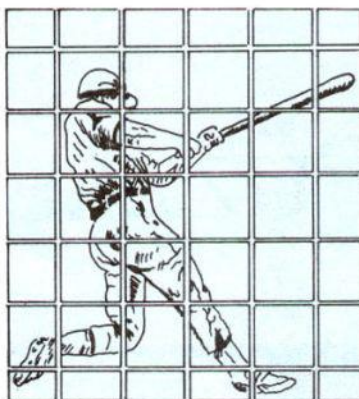
Scopo del gioco è di ricostruire questa figura nel minor tempo possibile.

Bisogna indicare al computer le coordinate della casella che vogliamo spostare e le coordinate di dove vogliamo metterla.

La casella lasciata vuota si riempirà del carattere presente nella nuova posizione. Cioè, se noi dobbiamo mettere il carattere che è nella casella A3 nella casella C2, il carattere presente in C2 verrà messo in A3, e viceversa.

Chi volesse sapere quante caselle sono al posto giusto, deve premere RETURN alla domanda "DA? **", ed il numero indicante le caselle che sono state messe al posto giusto verrà visualizzato in alto a sinistra.

Se si pensa di essere riusciti a ricostruire la figura, cioè se il numero di caselle giuste è 64, deve

ugualmente premere RETURN alla domanda di prima, e se non verrà verificato che è riuscito, verrà ugualmente visualizzato il numero di caselle che sono al posto giusto e tornerà in gioco, mentre in caso di vittoria ricomincerà il gioco con una nuova figura da ricomporre.

In totale il numero di figure è 2, ma se ne possono realizzare quante se ne vogliono, con piccole modifiche al programma, ed inserendo ovviamente i dati relativi alla nuova figura nelle istruzioni DATA.

Questi dati sono il valore dei PEEK da video dei caratteri che compongono la figura. L'unica eccezione è lo spazio che avrebbe valore 32, ma a cui va dato il valore 96 (= spazio + shift). ▪

```
 0 WW=0
 1 DATA96,96,96,96,96,96,96,96
 2 DATA96,96,96,96,96,96,96,96
 3 DATA96,96,81,96,96,96,77,96
 4 DATA96,96,96,96,90,96,100,96
 5 DATA96,96,81,96,96,96,78,96
 6 DATA96,96,96,96,96,96,96,96
 7 DATA96,96,96,96,96,96,96,96
 8 DATA96,96,96,96,96,96,96,96
10 DATA96,96,78,46,96,78,46,46,96,78,88,88,78,101,
   101,101,96,77,96,88,77,103
20 DATA103,103,96,78,77,46,96,77,46,96,78,96,88,77,46,
   96,96,46,77,88,96,46,77
30 DATA46,46,96,85,77,46,88,46,77,96,46,
   73,75,77,46,46,46,46,96
31 REM************************
32 REM** PUZZLE PER CBM 64   **
33 REM**  E PER IL PET !!!   **
34 REM**  GIOVANNI  BELLU'   **
35 REM**  VIA GIARDINI 20    **
36 REM**  SEREGNO (MILANO)   **
37 REM**                     **
38 REM**  TEL. 0362/239580   **
39 REM************************
40 PRINT"[CLR]"
41 INPUT"VUOI LA FIGURA DI AIUTO";W$:IFW$<>"S"ANDW$<>"N"THENPRINT"[CLR]":GOTO41
42 PRINT"[CLR]"
50 A$="[...]":A=1147
51 REM*******************************
52 REM*PER VERSIONE PET PORRE A=32891*
53 REM*******************************
60 PRINT"[HOME]     ABCDEFGH"
70 PRINT"    ┌────────┐"
80 FORK=1TO8:PRINTK"|[RVS]         [OFF]|"K:NEXT
90 PRINT"    └────────┘"
95 PRINT"     ABCDEFGH"
```

```
96 IFW$="S"THENGOSUB2000
100 FORX=1TO8:FORY=1TO8:READA(X,Y):B(X,Y)=A(X,Y):POKEA+40*Y+X,A(X,Y)
110 NEXT:NEXT
111 IFW$="S"THENGOSUB3000
120 DEFFNA(Z)=INT(RND(1)*8+1)
130 FORX=1TO8:FORY=1TO8
140 XX=FNA(Z):YY=FNA(Z)
150 IFB(XX,YY)=-1THEN140
155 C(X,Y)=B(XX,YY)
160 B(XX,YY)=-1:NEXT:NEXT
170 FORX=1TO8:FORY=1TO8:POKEA+40*Y+X,C(X,Y):NEXT:NEXT
180 FORX=1TO8:FORY=1TO8:B(X,Y)=C(X,Y):NEXT:NEXT
190 PRINTA$;:INPUT"PREMERE ■RETURN■ PER COMINCIARE    *■■■";T$
200 TI$="000000"
210 PRINTA$;"                                                  "
220 PRINTA$;:INPUT"DA     **■■■■";X$
225 IFX$="**"THEN1000
226 IFLEN(X$)>2THEN220
227 IFLEFT$(X$,1)<"A"ORLEFT$(X$,1)>"H"THEN220
228 IFRIGHT$(X$,1)<"1"ORRIGHT$(X$,1)>"8"THEN220
230 PRINTA$;:INPUT"A                ■■■■■■■■■■■     **■■■■";Y$
235 IFLEN(X$)>2THEN220
237 IFLEFT$(Y$,1)<"A"ORLEFT$(Y$,1)>"H"THEN230
238 IFRIGHT$(Y$,1)<"1"ORRIGHT$(Y$,1)>"8"THEN230
240 X=ASC(LEFT$(X$,1))-64
250 Y=VAL(RIGHT$(X$,1))
260 X1=ASC(LEFT$(Y$,1))-64
270 Y1=VAL(RIGHT$(Y$,1))
230 POKEA+40*Y+X,C(X1,Y1):POKEA+40*Y1+X1,C(X,Y):C=C(X1,Y1):C1=C(X,Y)
300 C(X1,Y1)=C1:C(X,Y)=C
400 GOTO210
1000 D=64
1010 FORX=1TO8:FORY=1TO8: IFA(X,Y)<>C(X,Y)THEND=D-1
1020 NEXT:NEXT
1030 IFD=64THENT$=TI$:PRINTA$;"HAI FINITO IN ";MID$(T$,3,2)" MINUTI E ";
1035 IFD=64THENPRINTMID$(T$,5,2)" SECONDI"
1036 IFD=64ANDW=0THENW=1:FORK=1TO2000:NEXT:PRINT"■":GOTO40
1037 IFD=64ANDW=1THENFORK=1TO2000:NEXT:PRINT"■■FINE GIOCO":END
1040 PRINT"■"D:GOTO210
2000 PRINT"■■■"TAB(20)"     ABCDEFGH"
2010 PRINTTAB(20)"    ┌────────┐"
2020 FORK=1TO8:PRINTTAB(20)K"|■          ■|"K:NEXT
2030 PRINTTAB(20)"    └────────┘"
2040 PRINTTAB(20)"     ABCDEFGH"
2050 RETURN
3000 FORX=1TO8:FORY=1TO8:POKEA+40*Y+X+20,A(X,Y)
3010 NEXT:NEXT
3020 RETURN
3030 REM REGOLO"_-_-_-_-_/_-_-_-_-_/_-_-_-_-_/_-_-_-_-_

READY.
```

STAMPA SU CARTA IN ALTA DEFINIZIONE

Un programma utile per tracciare funzioni matematiche.

Con il Vic 20 (o col Commodore 64), con la stampante MPS 801 (oppure la 1515), si possono realizzare grafici di funzioni definibili dall'utente.

La funzione va definita nella riga 170 come:

Y=F (X)* Y2 + Y2

subito dopo il RUN, il programma chiede alcuni dati necessari: scala dell'asse X (intervallo), scala asse Y, i due estremi dell'asse X.

Terminata l'acquisizione dei dati (righe 110-130), il computer immette nella variabile A (T) i valori delle potenze di 2 (da 210 a 216), valori che serviranno per disegnare il grafico.

La riga 150 assegna al contatore "PO" il valore 1 e ad A$ tanti CHR$(128) quanta lunghezza è stata assegnata alla richiesta della scala per l'asse Y.

La riga 160 controlla se è il momento di richiamare la subroutine per l'asse Y ed evita che,

quando X=0 la funzione venga calcolata.

Nella riga 170 c'è il cuore del programma: la funzione da disegnare.

Le due righe successive (180-190) fanno rientrare il disegno quando esce dai margini. Il blocco di righe dalla 200 alla 230 posiziona nella variabile A$ i punti del disegno e l'asse X.

La riga 250 controlla la fine del disegno. In caso affermativo c'è un END.

La riga 260 aggiorna il valore di "PO".

L'esecuzione riprende dalla riga 150.

La 270 chiude il ciclo tornando alla 160.

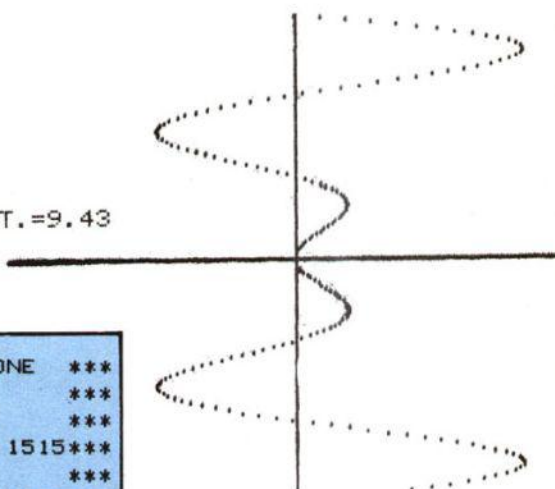
Il sottoprogramma che va da 1000 a 1020 serve per la stampa dell'asse Y e viene richiamato solo dalla riga 160.

Questo programma non può essere utilizzato con la stampante 1526, ma costituisce un valido suggerimento per adattarlo ad essa. ▪

Carlo Bernardi

```
Per ottenere questo grafico
inserite la riga seguente:

170 Y=(SIN(X)*X/10)*Y2+Y2

Alle domande rispondete cosi':
X=10 Y=127 EST.SIN.=-9.43 EST.DEST.=9.43
```

```
10 REM ***  STAMPA IN ALTA DEFINIZIONE   ***
20 REM ***                               ***
30 REM ***      VIC 20  COMMODORE 64     ***
40 REM ***STAMPANTI: MPS 801 OPPURE 1515***
50 REM ***                               ***
60 REM *** CARLO BERNARDI TEL.0521/54665***
70 REM ***      VIA MAZZOLA BEDOLI, 2    ***
80 REM ***           43100  PARMA        ***
90 :
100 OPEN4,4:GR$=CHR$(8)
110 INPUT"SCALA ASSE X 1-100";X1:IFX1>100ORX1<1THEN110
120 X1=X1/100:PRINT"INTERVALLO X1:";X1:INPUT"SCALA ASSE
    Y 1-127";Y2:IFY2<1ORY1>127THEN120
130 INPUT"ESTREMO SINISTRO X";X:INPUT"ESTREMO DESTRO X";X2:
    IFX>=X2THEN130
140 Y1=Y2*2:FORT=1TO7:READA(T):NEXT:DATA1,2,4,8,16,32,64
150 A$="":PO=1:FORT=1TOY1:A$=A$+CHR$(128):NEXT
160 IFPV<0ANDX=>0THENGOSUB1000:GOTO200
165 :
170 Y=(COS(X)/LOG(X))*Y2+Y2
175 :
180 IFY>Y1THENY=Y-Y1:GOTO180
190 IFY<0THENY=Y+Y1:GOTO190
200 SI$=LEFT$(A$,Y):CE$=MID$(A$,Y+1,1):DE$=RIGHT$(A$,Y1-LEN(SI$+CE$))
210 PR=ASC(CE$):TT=PR+A(PO):IFTT>255THENTT=255
220 CE$=CHR$(TT):A$=SI$+CE$+DE$:IFA(PO)=0THENA(PO)=PS
230 SI$=LEFT$(A$,Y2):CE$=CHR$(255):DE$=RIGHT$(A$,Y1-LEN(SI$+CE$))
    :A$=SI$+CE$+DE$
240 PV=X:X=X+.00002:X=X+X1:X=INT(X*1000)/1000
250 IFX>X2THENPRINT#4,CHR$(15)TAB((80-Y1/6)/2)GR$A$CHR$(15):CLOSE4:END
260 PO=PO+1:IFPO=8THENPO=1:PRINT#4,CHR$(15)TAB((80-Y1/6)/2)
    GR$A$CHR$(13)CHR$(15);:GOTO150
270 GOTO160
1000 FORT=1TOY1:SI$=LEFT$(A$,T-1):CE$=MID$(A$,T,1):DE$=RIGHT$(A$,
     Y1-LEN(SI$+CE$))
1010 PR=ASC(CE$):TT=PR+A(PO):IFTT>255THENTT=255
1020 CE$=CHR$(TT):A$=SI$+CE$+DE$:NEXT:PS=A(PO):A(PO)=0:RETURN
READY.
```

La Casa del Mostro

Il programma gira su Commodore 64 e su Vic 20 anche se inespanso, ed è un libero adattamento da un listato in basic Microsoft.

Il gioco consiste nello sfuggire il "mostro", che si muove verso il giocatore mediante i comandi NORD, SUD, EST, OVEST che, essendo controllati da un GET, diventano N, S, E, O.

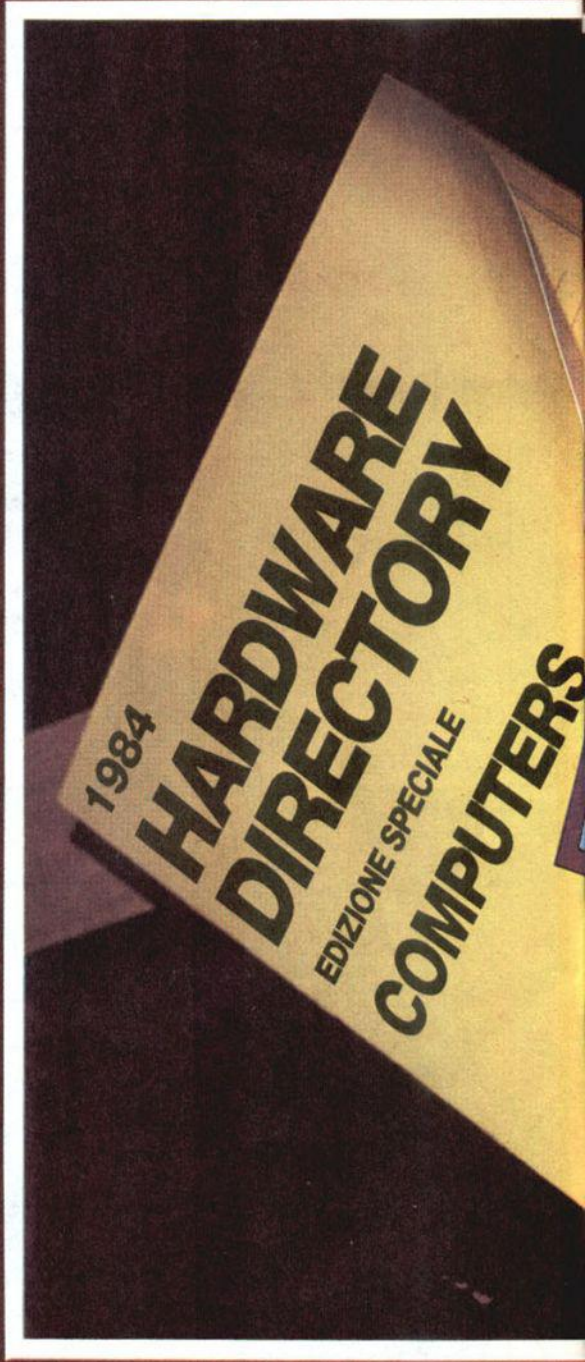
Tutto il gioco è controllato in modo che non c'è bisogno di premere altri tasti oltre i comandi e si conclude comunque entro la decima mossa. Risulta difficile sfuggire il mostro, in quanto il suo movimento non è casuale, ma guidato sulla posizione del giocatore da un'apposita routine.

Il giocatore ha la possibilità di fare salti di due caselle, anteponendo un "2" al comando; allo stesso tempo, però, mentre il giocatore può muoversi solo in orizzontale o in verticale, il mostro può muoversi anche diagonalmente. Si precisa che il mostro fa la sua mossa prima di quella del giocatore. ■

COMPUTER

il "NEWSMAGAZINE" dell'informatica

COMPUTER

il "NEWSMAGAZINE" dell'informatica

30
Hobby
& Home
Computer
350
mini
e
micro
150
Mainframe
COMPUTER
PROGRAMMARE
PRESTO E BENE
In omaggio
il volume

```
100 REM-----------------------
110 REM ** CASA DEL MOSTRO **
120 REM
130 REM VIC 20 & COMMODORE 64
140 REM-----------------------
150 REM ** MARCO MANTOVANI **
160 REM
170 REM ** TEL.051/387156   **
180 REM-----------------------
190 :
200 PRINT"■********************"
210 PRINT"** CASA DEL MOSTRO **"
220 PRINT"********************"
230 GOSUB880:PRINT"■"
240 X=3:Y=1
250 R=3:C=5
260 FORT=1TO10
270 PRINT"■";
280 FORI=1TO5
290 FORJ=1TO5
300 PRINTSPC(1);
310 IFI=XANDJ=YTHENPRINTTAB(11)
    "M";:GOTO340
320 IFI=RANDJ=CTHENPRINTTAB(11)
    "X";:GOTO340
330 PRINTTAB(11)"●";
340 NEXTJ
350 PRINT
360 NEXTI
370 PRINT"■■■ BATTUTA NUMERO ";T
380 PRINT"■DIREZIONE (N,S,E,O)?"
390 PRINT"OPPURE
    (2N,2E,2S,2O)":N$=""
400 GETM$
410 IFM$="2"THENN$=M$:GOTO400
420 IFM$<>"N"ANDM$<>"E"ANDM$<>
    "S"ANDM$<>"O"THEN400
430 IFN$="2"THENM$=N$+M$
440 PRINT"■■■■■■■"
    :PRINTTAB(33)"■ "M$" "
450 IFM$="N"THENR=R-1
460 IFM$="2N"THENR=R-2
470 IFM$="E"THENC=C+1
480 IFM$="2E"THENC=C+2
490 IFM$="S"THENR=R+1
500 IFM$="2S"THENR=R+2
510 IFM$="O"THENC=C-1
520 IFM$="2O"THENC=C-2
530 IFR*C>0ANDR<=5ANDC<=5THEN580
540 PRINT"■ FUORI GRIGLIA
    ■**IGNORANTE!**"
550 PRINT"SE NON CONOSCI"
560 PRINT"I PUNTI CARDINALI"
570 PRINT"■COMPRATI
    UNA BUSSOLA":GOTO840
580 IFR<>XORY<>CTHEN600
590 PRINTSPC(13)"■■
    MANGIATO!!!  ":GOTO840
600 PRINT
610 IFX=RANDY<CTHEND=1
620 IFX>RANDY<CTHEND=2
630 IFX>RANDY=CTHEND=3
640 IFX>RANDY>CTHEND=4
650 IFX=RANDY>CTHEND=5
660 IFX<RANDY>CTHEND=6
670 IFX<RANDY=CTHEND=7
680 IFX<RANDY<CTHEND=8
690 IFD=0THEND=8
700 IFD=9THEND=1
710 IFD>1ANDD<5THENX=X-1
720 IFD>5THENX=X+1
730 IFD>3ANDD<7THENY=Y-1
740 IFD<3ORD=8THENY=Y+1
750 IFX=0THENX=1
760 IFY=0THENY=1
770 IFX=6THENX=5
780 IFY=6THENY=5
790 IFX<>RORY<>CTHEN810
800 PRINTSPC(13)"■■ MANGIATO!!!
    ":GOTO840
810 NEXT
820 PRINT"■                        "
830 PRINT"■ SEI SOPRAVISSUTO! "
840 PRINT"GIOCHI ANCORA (S/N) ?"
850 GETY$:IFY$="S"THENRUN
860 IFY$="N"THENEND
870 GOTO850
880 PRINT"IL GIOCO CONSISTE"
890 PRINT"NELLO SCAPPARE DAL"
900 PRINT"MOSTRO (M) CHE SI"
910 PRINT"MUOVERA' VERSO DI"
920 PRINT"TE (X) PER MANGIARTI."
930 PRINT"SE RIUSCIRAI A"
940 PRINT"SFUGGIRGLI PER 10"
950 PRINT"MOSSE SARAI SALVO."
960 GETA$:IFA$=""THEN960
970 RETURN

READY.
```

TENNIS PER DUE

```
100 REM <><><><><><><><><><><><><><><>
102 REM <>   (PER DUE GIOCATORI)     <>
104 REM <>    TENNIS PER CBM 64      <>
106 REM <>                           <>
108 REM <>      EMANUELE BALDI       <>
110 REM <>                           <>
112 REM <>     TEL.  039/745184      <>
114 REM <>JIOYSTICK OPP. TASTI >PXE <>
116 REM <><><><><><><><><><><><><><><>
118 REM  --------------------------
120 REM   ↑↑↑↑↑↑↑↑↑  INPUT  ↑↑↑↑↑↑↑↑↑
122 REM  --------------------------
124 INPUT"SEI UN MERLO ";A$:IFA$=
    "S"THENPOKE53248+23,3:P1=0:GOTO130
126 INPUT"RACCHETTA (0/2)";
    P1:POKE53248+23,0
128 INPUT"SEI UN MOSTRO ";A$:IFA$=
    "S"THENQ=15:C=-20:GOTO132
130 INPUT" VELOCITA'  (6/10)";C:Q=
    C-1:C=-C:IFC=-1THENQ=1
132 INPUT"CON 2 JOISTYCK";A$:IFA$=
    "S"THEN137
134 INPUT"CON 1 JOISTYCK";A$:IFA$="S"THEN138
136 W1=14:W2=23:W3=41:W4=44:L5=203:L6=L5:GOTO139
137 W1=126:W2=125:W3=254:W4=253:L5=56320:L6=
    56321:GOTO139
138 W1=126:W2=125:W3=41:W4=44:L5=56320:L6=203
139 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,60,0,0,129,0,
    0,255,0,0,255,0
140 DATA0.255,0,0,255,0,0,129,0,0,60,0,0,0,0,0,0,0,0,0,0,0,0,
    0,0,0,0,0,0,0,0
142 DATA0,0,-1,0
144 DATA0,255,128,0,255,128,0,255,128,0,255,128,0,255,128,0,255,
    128,0,255.128
146 DATA0,255,128,0,255,128
148 DATA0,255,128,0,255,128,0,255,128,0,255,128,0,255,128,0,255,
    128,0,255,128
150 DATA0,255,128,0,255,128
152 DATA0,255,128,0,255,128,0,255,128,-1
154 FORR=704TO704+63:POKER,0:NEXT
```

```
156 PRINT"■":V=53248:POKEV+21,7:POKE2040,11:POKE2041,
    11:POKEV+39,1:POKEV+40,1
158 POKEV,45:POKEV+2,238
160 POKE2042,13:POKEV+41,1:PRINTSPC(33)" SCORE"
162 REM  ----------------------------
164 REM   ↑↑↑↑↑↑↑  SCHERMO  ↑↑↑↑↑↑↑↑↑
166 REM  ----------------------------
168 PRINT"■━━━━━━━━━━━━━━━━━━━━━━━━━━━━■■■":A$="■ "
170 FORR=0TO20:PRINTA$:NEXT
172 PRINT"■━━━━━━━━━━━━━━━━━━━━━━━━━━━━■■■":PRINT"■"
174 A$="■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ |"
176 FORR=0TO21:PRINTA$:NEXT
178 POKE1065,79:POKE1095,80:POKE1345,76:POKE1975,122
180 A$="■■■■■■■■■■■■■■■■■■| ":PRINT"■"
182 FORR=0TO22:PRINTA$:NEXT:PRINT"■"
184 PRINT"■■■■■■■■■■■■■■■■■■■━━━━━━━━━━━━━━━━━━"
186 DX=DX+1:READDA:IFDA=-1THEN190
188 POKE831+DX,DA:GOTO186
190 FORP=0TOP1*21:READP2:NEXT
192 DY=DY+1:READDA:IFDA=-1THEN196
194 POKE703+DY,DA:GOTO192
196 POKEV+4,60:POKEV+5,120:PX=80:PY=125:Y=135:YY=135:PRINT"■"
198 REM  ----------------------------
200 REM   ↑↑↑↑↑↑↑  GIOCO   ↑↑↑↑↑↑↑↑↑
202 REM  ----------------------------
204 IFFL=0THENGOTO216
206 IFCC=1THENCC=0:GOTO234
208 A=PEEK(L5):FL=0
210 IFA=W1THENY=Y-Q:GOTO214
212 IFA=W2THENY=Y+Q
214 POKEV+1,Y:GOSUB224
216 B=PEEK(L6):FL=1
218 IFB=W3THENYY=YY-Q:GOTO222
220 IFB=W4THENYY=YY+Q
222 POKEV+3,YY:GOSUB224:GOTO204
224 IFPEEK(V+30)<>0THENC=-C:D=RND(1)*2-RND(1)*2:POKEV+4,PX+C:
    POKEV+30,0:GOTO228
226 GOTO230
228 POKE54296,65:POKE54296,0:RETURN
230 PX=PX+C:PY=PY+D:IFPX>242ORPX<25ORPY>230ORPY<50THENCC=1:RETURN
232 POKEV+4,PX:POKEV+5,PY:RETURN
234 IFPX<137THENS2=S2+1:IFS2>(S1+1)ANDS2>15THENPOKEV+21,0:GOTO240
236 IFPX>137THENS1=S1+1:IFS1>(S2+1)ANDS1>15THENPOKEV+21,0:GOTO244
238 PRINTSPC(32)S1"■-"S2:WAIT203,32:GOTO196
240 PRINT"■ -A- HAI VINTO !!!"
242 PRINT"■■■■■SCORE■"S1"-"S2:END
244 PRINT"■ -B- HAI VINTO !!!"
246 PRINT"■■■■■SCORE■"S1"-" S2:END
248 REM REGOLO "_‾_‾_‾_‾_‾/_‾_‾_‾_‾_‾/_‾_‾_‾_‾_‾/_‾_‾_‾_‾_‾

READY.
```

Guida mercato Commodore

Prodotto — **Prezzo (IVA esclusa)**

VIC - 20

Home Computer Vic 20		199.000
Unità di espansione	*(1020)*	295.000
Modulo di espansione	*(1023)*	135.000
Cartuccia da 3K di memoria	*(1210)*	49.000
Cartuccia da 8K di memoria	*(1110)*	75.000
Cartuccia da 16K di memoria	*(1111)*	125.000
Cartuccia Vic rel	*(4011)*	95.000

Permette di controllare il funzionamento di allarmi antifurto, porte automatiche, telefoni, trasmittenti ed apparecchi similari.

Vic switch *(4012)* 225.000

Possono essere collegati fino a 16 VIC 20 con un floppy e una stampante (distanza massima 1500 mt.).

Interfaccia IEEE 488	*(T-1)*	175.000
Interfaccia centronics	*(T-3)*	115.000
RS232-C adapter	*(1011-A)*	75.000
RS132-C adapter	*(1011-B)*	75.000

Commodore 64

CPU 64K RAM *(CBM64)* 625.000

C 64 Executive *(SX 64)* 2.350.000

Sistema operativo CP/M *(CP/M)* 125.000

Consente di programmare il Commodore 64 in linguaggio CP/M, il più utilizzato sui Personal Computers. Permette inoltre di accedere alla enorme libreria di Software applicativi CP/M.

Pet speed *(6411)* 95.000

Compilatore basic che aumenta la velocità di esecuzione dei programmi di circa 40 volte.

Accessori per Vic e Commodore 64

Stampante plotter a colori *(1520)* 375.000

80 caratteri, per linea, 4 colori, alla risoluzione di 0,2 mm per passo.

Unità stampante *(MPS 801)* 450.000

Stampa velocemente su carta normale quanto appare sul video: programmi, lettere, dati, grafici.

Unità stampante *(1526)* 655.000

Stampante 80 colonne, bidirezionale, 60 CPS, spaziatura programmabile, trazione a frizione o a trattore.

Registratore dedicato *(1530)* 120.000

Per memorizzare facilmente programmi e dati su normali cassette magnetiche.

Floppy disk drive *(1541)* 630.000

Veloce unità di memoria di massa ad alta capacità. Può immagazzinare fino a 170.000 caratteri su ogni singolo disco.

Monitor monocromatico *(1601)* 285.000

A fosfori verdi 12".

Per giocare

Comando per giochi (Joystick) *(1311)* 13.500

Permette di muoversi in tutte le direzioni, di iniziare i vari giochi di movimento e di "sparare".

Comando a manopola per giochi (Paddle) *(1312)* 22.500

Adatto per i giochi a 2 persone, esegue movimenti in orizzontale e verticale.

Commodore 4000

CPU 16K RAM *(CBM 4016)* 1.395.000

18K ROM, BASIC 4.0 residente, video 40 colonne per 25 righe, tastiera semigrafica.

CPU 32K RAM *(CBM 4032)* 1.495.000

18K ROM, BASIC 4.0 residente, video 40 colonne per 25 righe, tastiera semigrafica.

Commodore 9000

Doppia CPU 134K RAM *(CBM 9000)* 2.350.000

Micro Main Frame Computer a doppia CPU (6502 - 6809) compatibile con tutte le periferiche Commodore della serie 8000. Include 5 linguaggi di programmazione. (COBOL, FORTRAN, TCL PASCAL, UCSD PASCAL, APL).

Commodore 8000

CPU 32K RAM *(8032 SK)* 1.725.000

18K ROM, Basic 4.0 residente, video orientabile e basculante 80 colonne per 25 righe, tastiera commerciale separata.

CPU 96K RAM *(8096 SK)* 2.285.000

18K ROM, Basic 4.0 residente, video orientabile e basculante 80 colonne per 25 righe, tastiera commerciale separata. Include sistema operativo PM/96.

Commodore 600

Indicato per applicazioni industriali, collegamento a strumentazione, controllo numerico, ecc. Utilizza monitor in commercio.

CPU 128K RAM *(610)* 2.150.000

CPU 128K RAM espandibileinternamentea 256K e esternamente a 960K, interfaccia RS232C, IEEE 488, porta utente a 8 Bit. Compatibile con tutte le periferiche Commodore della serie professionale.

CPU 256K RAM *(620)* 2.550.000

CPU 256K RAM espandibili esternamente a 960K. Caratteristiche uguali al Mod. 610.

Monitor *(1601)* 285.000

Monocromatico a fosfori verdi, 12".

Commodore 700

CPU 128K RAM *(710)* 2.850.000

CPU 128K RAM espandibili internamente a 256K ed esternamente a 960K. Video orientabile e basculante 80 colonne per 25 righe. Compatibile con tutte le periferiche Commodore delle serie professionali.

COMMODORE 64

HOTLINE · UPDATE · GARANZIA

Tre nuove parole nel campo dell'informatica. Esse rappresentano il

- NUOVO SERVIZIO -

che la Leoni Informatica, prima fra tutti offre ai suoi clienti. **COMMODORE 64**

HOTLINE

— linea telefonica dedicata alla risoluzione dei problemi dei clienti. Chiamando il numero telefonico riservato che troverete sulla cartolina garanzia acclusa ai programmi, riceverete tutte le informazioni che vi necessitano.

UPDATE

— servizio di aggiornamento continuo dei programmi acquistati. Ogni modifica ai programmi realizzati dalla Leoni Informatica sarà fornita agli utenti degli stessi.

GARANZIA

— tutti i programmi Leoni Informatica sono coperti da garanzia a Vita contro guasti di origine.

ALCUNI PROGRAMMI PER COMMODORE 64

Cod.	Descrizione	Prezzo
0047	Gestione Anagrafiche	120.000
0050/c	Totocalcio Sviluppo Colonnare	80.000
0051/c	Gestione dei conti Casa	100.000
0055/c	Impariamo il basic	100.000
0056/c	Dichiarazione Iva	60.000
0063/c	Cento Programmi per il 64	80.000
0064	Compilatore Pet Speed	80.000
0065/c	Fido Clienti	100.000
0066	Conto Corrente	150.000
0067	Gestione piano dei Conti	150.000
0068	Gestione Appuntamenti	150.000
0071	Gestione Ordini	150.000
0086	Gestione Librerie	150.000
0090	Mailing List	150.000
0091	Rubrica Telefonica	120.000
0094/c	Gestione Scheda 4800 car.	160.000
0096	Gestione Scheda Agganciata al Mailing List	250.000
0116	Scadenziario Effetti	200.000
0120	Contabilità Fatture Iva/Imponibile	200.000
0121	Contabilità Semplice	400.000
0136	Legge 373	150.000
0143	Magazzino Grossisti (2500 art.)	380.000
0144	Magazzino Fatturazione Agganciate	400.000
0148	Gestione Ottici	300.000
0149	Gestione Dentisti	300.000
0151	Gestione Farmacie	400.000
0152	Gestione Studi Medici	300.000
0158	Magazzino Dettaglio (2500 art.)	380.000
0159	Magazzino Taglia e Colore	380.000
0160	Bolle e Fatture	300.000
0162/c	Screen Grafix	150.000
0163	Copia Disco Singolo	50.000
0165/c	Assembler Disassembler	80.000
0169/c	Magazzeno alfanumerico (1100 articoli)	250.000
0080	Gestione Clubs Nautici	250.000
0081	Gestione Officine	400.000
0087	Gestione Ristoranti	400.000
0131	Gestione Hotel/Pensioni	400.000
0132	Gestione Parrucchieri	400.000
0133	Gestione Gommisti	400.000
0170	Gestione Tavola Calda	400.000
0171	Gestione Lavanderia	400.000
0155	Gestione Condominio	300.000
0166/c	Compactor	50.000
0167/c	Scompactor	50.000
0168/c	PET Emulator	35.000
0306/c	Character editor	40.000
0309/c	Hires image	40.000
0310/c	Hard copy	40.000
0312	Master 64	225.000
0313/c	Tool 64	85.000
0314/c	Stat 64	65.000
0157	Calc Result Advanced	350.000
0319	Easy Script	125.000
0322/c	Forth 64	65.000

I programmi Leoni sono disponibili presso tutti i punti vendita MELCHIONI ELETTRONICA
Vendita per corrispondenza anche dell'HARDWARE

20142 MILANO - VIA DON RODRIGO, 6 - TEL. 02/8467378

$ 175.000 IN PALIO

CREATE A NEW VIDEO GAME AND YOU COULD

MAKE YOURSELF A

MILLIONAIRE

Esclusivo per i lettori di Commodore Computer Club

IRP

International Video Game of the Year

Crea un videogioco intelligente ed originale e potrai diventare milionario. Questo fantastico concorso, organizzato dall'IRP (The International Register of Indipendent Computer Programmers ltd.) e dal famoso Marc McCormack International Management Group, e sponsorizzato per l'Italia da Commodore Computer Club, ti offre un'occasione unica nella vita. I premi sono elevati ed immediati, e ad essi si aggiungerà una royalty del 10% sulle vendite dei giochi premiati ai più importanti distributori in tutto il mondo. I vincitori, inoltre, potranno partecipare ad una serie di trasmissioni sulle principali reti televisive del mondo. La tua conoscenza dei computers e la tua immaginazione possono farti diventare rapidamente ricco e famoso!

PRIMO PREMIO: $ 100.000 *OLTRE* CINQUE PREMI DI CONSOLAZIONE DI $ 15.000

Prepara un videogame nuovo e originale per una delle seguenti categorie: SPORT, SIMULAZIONI, ARCADE, STRATEGIA, AVVENTURA/FANTASIA o per la sezione speciale prevista per quei programmi che, pur non essendo dei veri giochi, abbiano notevoli caratteristiche didattiche o di divertimento. Sono altresì previsti una serie di premi al "Merito" che consentiranno di stampigliare sulle confezioni commerciali dei videogiochi selezionati la dizione "An International Videogame of The Year MERIT AWARD".
E' una grande sfida. Ed i premi, sia in termini finanziari che di prestigio, sono eccezionali.
Questo è certamente il concorso internazionale più eccitante per ogni appassionato di computer.

APPARIRETE IN TV!

I sei vincitori saranno invitati a far parte di uno spettacolare show televisivo che verrà distribuito ai più importanti network del mondo. Quanto basta per renderti famoso!

COME PARTECIPARE

Invia il tuo gioco o i tuoi giochi su cassetta indicando su quale computer gira o, girano, utilizzando questo coupon di qualificazione. Riceverai anche una documentazione completa con le regole dettagliate del concorso.

ENTRO E NON OLTRE IL 31 MAGGIO 1984

A: IRP Limited, Pinewood Film Studios, Iver, Bucks England

Nome: ____________

Indirizzo ____________

SC1